Treball de Fi de Grau/Màster

Grau en Enginyeria en Tecnologia Industrial

Estudi de la Viabilitat sobre Reparació en l'Electrònica: Elaboració de l'IC Tester

MEMÒRIA

Autor: Cristian Gimeno Peredo
Director: Cristina Lampón Diestre

Convocatòria: 01 - 2020

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Resum

L'objectiu del treball exposat a continuació és el de construir un testejador de circuits integrats anomenat IC-Tester, una eina molt útil per a la reparació de circuits electrònics.

La gran majoria d'aparells electrònics disposen de circuits integrats *Integrated Circuits o IC* en anglès, també anomenats xips que desenvolupen un seguit d'operacions fixades. Aquestes operacions són intrínseques a la seva estructura física i són predeterminades en la construcció dels xips.

Els xips acostumen a ser implementats sobre plaques base juntament amb altres components com resistències, condensadors, díodes... Les connexions amb la resta de dispositius es fan mitjançant "pistes" de material conductor en comptes de cablejat. Aquestes pistes són construïdes sobre la placa en el moment de la seva fabricació. Degut al gran nombre de components, xips i pistes, la reparació de dispositius electrònics pot esdevenir una tasca llarga, tediosa i inclús infinita si no es disposa de les eines o coneixements.

A més de la gran varietat d'eines necessàries per estudiar i entendre el funcionament del dispositiu electrònic a reparar, els coneixements electrònics que tingui la persona reparadora poden no ser suficients per a identificar i solucionar el problema. La naturalesa d'aquesta insuficiència recau en dues problemàtiques:

- Xips i components no estan identificats ni etiquetats.
- El disseny de la placa base no ha sigut compartit pel fabricant.

Són aquestes les dues tendències pròpies del disseny i fabricació a les que aquest treball pretén fer front.

L'elaboració de l'IC Tester es troba fonamentada en dos pilars. E primer lloc, el disseny de funcionalitat del testejador mitjançant el programari de codi obert d'Arduino. Aquest codi requerirà del lector ser coneixedor de programació informàtica per l'enteniment complert del disseny. En segon lloc, la construcció del testejador mitjançant una placa d'Arduino i components simples d'electrònica els quals només requeriran coneixements bàsics.

Un cop llegit i entès aquest document, el lector serà capaç d'elaborar el seu propi testejador mitjançant la guia adjunta que indica pas a pas el procediment de construcció implementat per fabricar l'IC-Tester.

Sumari

1. Prefaci

1.1. Origen del projecte

L'electrònica, nascuda com a branca de la física aplicada amb el descobriment de l'electró al 1897 [1], ha sigut, és i serà un dels pilars fonamentals sobre el qual es desenvolupa l'enginyeria.

Com a conseqüència del ràpid progrés de la tecnologia moderna i la informàtica, el camp de l'electrònica ha esdevingut objecte d'estudi i d'innovació, sempre tendint cap a tres objectius principals: l'augment de funcions, la millora en l'eficiència de funcionament i la minimització del cost i espai necessari.

Aquesta gran varietat de funcions en un espai tan reduït posiciona als components electrònics en l'avantguarda d'instrumentació necessària per a la fabricació de productes elèctrics que requereixin controls complexos, impossibles d'implementar mecànicament.

Així doncs l'electrònica avança cada cop més en l'adquisició del control global del dispositiu o sistema fins al domini total i és en aquesta dependència on s'origina la fragilitat del conjunt, ja que si un petit component electrònic falla, tot el mecanisme es dona per invàlid.

Aquesta fallida pot ser entesa de dues maneres, les dues visions relacionen la fallida del component a la fi de la seva vida útil, la qual es dona per una degradació dels materials que el conformen o un mal ús que exigeix el sobreesforç del dispositiu escurçant el cicle vital. No obstant la primera visió, més comuna i generalitzada, atribueix la longitud de la vida útil a una conseqüència de l'estat actual de les tecnologies i materials emprats en la fabricació, els quals treballen fins a exhaurir les seves capacitats. La segona visió relaciona l'escurçament de la vida útil amb un disseny premeditat amb l'objectiu que el dispositiu falli abans del que és capaç. Aquesta tendència a disseny per fallar s'anomena Obsolescència Programada.

1.2. Motivació

És de la lluita contra l'obsolescència programada i el dret a disposar de productes duradors i fiables d'on neix la motivació per realitzar aquest projecte. Motivació que intenta facilitar i apropar a un ampli rang d'usuaris a la reparació de l'electrònica, els quals bé per insuficiència de coneixements o d'instrumentació, atribueixen aquest tipus de reparacions a una tasca lenta i difícil.

Un cop trobada la motivació en la millora de la reparació d'electrònica, s'ha de focalitzar en els entrebancs que aquesta proposa. La principal dificultat que comporta la majoria de temps de reparació és desconèixer on es troba la fallida del dispositiu electrònic. L'error es pot donar en una connexió entre components o en el component mateix.

La primera fallida, en les interconnexions, és de relativa fàcil identificació verificant la continuïtat del corrent entre components. La segona fallida, intrínseca al component, és molt més complexa donat que cada component disposa d'un funcionament propi, el qual si no es coneix pot ser impossible de deduir. La desconeixença del funcionament d'un component pot donar-se per dues raons: la primera, el component no es troba etiquetat ni existeix cap referència i la segona, que es dona quan l'esquemàtic del circuit electrònic amb els components no ha sigut compartit pel fabricant del dispositiu.

Els components electrònics més bàsics com els condensadors o resistències venen etiquetats seguint codis propis que indiquen els valors nominals, límits, toleràncies... No obstant, altres components més complexos com els circuits integrats sovint no disposen d'etiquetatge ni referència. Aquests circuits, explicats amb deteniment al llarg del document, són els majors responsables del correcte funcionament del dispositiu i si fallen, el dispositiu quedarà inactiu. L'usuari, desconeixent el funcionament d'aquest component en l'intent de reparar, no podrà verificar el seu estat, impedint la identificació de la problemàtica.

És per aquesta raó que la millora en la reparació que motiva aquest projecte troba el seu focus en la identificació i verificació dels circuits integrats únicament.

És de gran importància destacar la reducció de l'impacte ambiental negatiu com a motivació addicional d'aquest projecte. Qualsevol dispositiu electrònic que es pugui reparar i tornar a utilitzar afectarà positivament i de bon grau les grans problemàtiques que atempten contra el benestar ambiental.

Al reparar el dispositiu no serà necessària l'extracció de nous recursos per suplir amb un nou producte les necessitats que l'anterior ja resolia, a més, els materials podran utilitzar-se fins exhaurir la seva veritable vida útil. Al disminuir la generació de nous dispositius electrònics s'aconseguirà també influir en la reducció de residus.

2. Introducció

2.1. Objectius del projecte

L'objectiu genèric d'aquest projecte és el d'elaborar una eina que doni suport a la tasca de reparació en l'electrònica. Pels motius presentats en apartats anteriors es fabricarà un testejador de circuits integrats. El procés proposa tres objectius específics representatius de cada fase.

La primera etapa busca assolir l'objectiu de dissenyar el funcionament del dispositiu combinant les plataformes de software i hardware. S'estudiaran els modes i funcions del testejador així com els components electrònics necessaris per la implementació.

La segona fase, de construcció, té com objectiu materialitzar el disseny prèviament creat mitjançant la placa base d'Arduino, el cablejat i tots els components electrònics. La construcció es descompondrà en dues parts, una primera de prototipatge on s'analitzarà si el disseny teòric s'adequa a la realitat, i la segona, la construcció definitiva del testejador.

L'últim objectiu específic que contempla aquest projecte és el de l'elaborar una guia de muntatge del testejador, per a què el lector pugui confeccionar la seva pròpia eina.

2.2. Abast del projecte

A nivell divulgatiu, el projecte, o més aviat la guia de construcció, es portarà a concurs. El concurs és organitzat per la plataforma web Instructables [2] dedicada a la publicació de guies de fabricació de productes en totes les disciplines.

3. DISSENY

3.1. INTRODUCCIÓ ALS CIRCUITS INTEGRATS

3.1.1. Definició i classificació

Els circuits integrats també coneguts com xips o microxips són unes estructures de material semiconductor que desenvolupen unes operacions fixades en la seva fabricació. Aquestes estructures s'allotgen en encapsulaments de material polimèric que aïllen els semiconductors, però permeten la connexió dels circuits interiors amb l'exterior, a través de pins o potes que traspassen l'encapsulament.

Els circuits es classifiquen en tres grans grups segons el tipus de variables amb les que treballen: poden ser digitals, analògics o mixtos. Les variables digitals són discretes, només poden prendre un determinat nombre de valors que aplicat a l'electrònica són 0 o 1. Les variables analògiques són contínues, poden adoptar infinits valors dins d'un rang sempre i quan la resolució ho permeti. Els circuits integrats mixtos són aquells que operen amb ambdues variables.

Generalment, en els sistemes d'instrumentació electrònica es capta la informació del món real, de naturalesa analògica, a través de sensors. Aquests senyals es converteixen en digitals mitjançant convertidor A/D, es processa la informació en els circuits interns i finalment s'envia la resposta digital desitjada a l'exterior a través d'actuadors, un cop convertida a analògica.

Pel que fa al projecte, de tota la cadena de transmissió interessa el processament d'informació, el qual degut a la simplicitat de variables s'acostuma a realitzar en el món digital. És per aquesta raó i per la gran complexitat de treball amb variables analògiques que l'objectiu que proposa aquest projecte només serà aplicable a circuits integrats digitals.

3.1.2. Series 7400 i 4000

Per tal de poder comptar amb una base de dades que permetés desenvolupar i depurar l'IC Tester, s'ha decidit començar amb un nombre reduït de circuits integrats. Així doncs, s'apliquen dues simplificacions sobre el disseny del testejador. La primera, ja coneguda, és enfocar l'IC Tester només a circuits integrats digitals. La segona, que capitula aquest apartat, és reduir tot el conjunt de xips digitals a les sèries de portes lògiques 7400 i 4000. En el cas d'aquestes sèries s'han obtingut dos tipus de documents molt valuosos:

El primer tipus consta de totes les *datasheets* de cada IC per separat [3] [4]. La *datasheets* o fitxa de dades, ampliat a tots els components electrònics, es podria considerar com el carnet d'identitat del component. En aquestes fitxes s'engloba tota la informació pertinent a l'estructura, implementació i funcionalitat del component. Pel que fa a la relació dels circuits integrats amb aquest projecte, la informació imprescindible a extraure de les *datasheet* és el tipus d'encapsulat, la taula de veritat explicada a continuació i l'esquema de pins per realitzar les connexions pertinents i amb quins valors de voltatge o intensitat.

El segon document [5] trobat respecte les sèries 7400 i 4000 està composat per totes les taules de veritat dels IC comprimides en un únic arxiu, informació molt útil per algunes funcions del testejador.

3.1.3. Taula de veritat

La taula de veritat d'un IC és la informació més útil un cop el xip es troba en funcionament, ens indica els valors que tindran totes les sortides depenent de les entrades que introduïm, per cada un dels respectius pins. Les entrades són anomenades variables i es classifiquen com a "Input", les sortides s'anomenen estats i són "Output". Cal distingir, però, entre la taula de veritat i l'esquema de pins. Aquest últim, apart de les entrades i sortides també ens indica quins pins han de estar connectats al voltatge d'alimentació i amb quina polaritat.

La informació obtinguda de les *datasheet* dels xips és una taula de veritat ampliada als esquemes de pins. Existeixen set possibles valors que poden prendre els pins dins d'aquesta taula de veritat, són:

- V: voltatge d'entrada.
- G: "ground"/ terra / 0V.
- 1: entrada de valor 1 / alt / Input.
- 0: entrada de valor 0 / baix / Input.
- H: sortida de valor 1 / alt / Output.
- L: sortida de valor 0 / baix / Output.
- C: "clock", rellotge. Variable d'entrada que provoca al pin oscil·lar entre valor alt / 1 i baix / 0. Sovint necessari per actualitzar l'efecte de les variables d'entrada sobre els estats de sortida o sincronitzar-se.

3.2. ESTRUCTURA I FUNCIONAMENT DEL TESTEJADOR

3.2.1. Estructura

Seguint un punt de vista funcional, el testejador està composat per sis blocs. Aquests blocs interactuen entre si mitjançant la placa base d'Arduino, la qual composa el cos central, el cervell del nostre dispositiu. Tot el programa de software estarà guardat en aquest processador. Aquest enviarà i rebrà senyals per coordinar i controlar els següents sistemes, els qual mai es connectaran entre si obviant el processador. El llistat de blocs funcionals ha estat ordenat segons aparició en l'ús del testejador.

· Computador: ordenador mitjançant el qual guardem el programa de software sobre el processador. A més s'utilitzarà per gravar les taules de veritat de cada IC en la memòria portàtil explicada a continuació.

· EEPROM : *electrically erasable programmable read-only memory*, memòria on emmagatzemarem les taules de veritat dels ICs que consultarem quan testegem els circuits. És important que aquesta memòria sigui programable elèctricament ja que no volem que la informació estigui fixada de fàbrica sinó modificar-la contínuament al afegir noves taules de veritat. La capacitat de la memòria emprada é de 256K que equival a 32K x 8 bits. A cada adreça d'informació, cavitat física on es guarden els valors, es poden gravar fins a 8 bits i existeixen 32000 adreces, totes elles reprogramables 1.000.000 cops de manera fiable.

· Display: pantalla, interfície que permetrà a l'usuari ser coneixedor dels esdeveniments principals que s'estan duent a terme al testejador, així com poder escollir el mode de funcionament i visualitzar els efectes del control.

· Control: mecanisme mecànic, interruptors de botons que ens permetran moure'ns pel display, accedir als modes de treball i introduir la informació que es requereixi.

· Pinça: és el sistema que connecta el processador amb l'IC a testejar. A través d'aquesta connexió s'enviaran els senyals de sortida, Inputs, i es rebran els d'entrada, Outputs.

La garantia d'èxit del dispositiu recau en la correcta coordinació i reunió entre els blocs funcionals els quals s'implementaran de manera diferent depenent del mode de funcionament. A la Figura 1 es pot apreciar l'esquema de l'estructura.

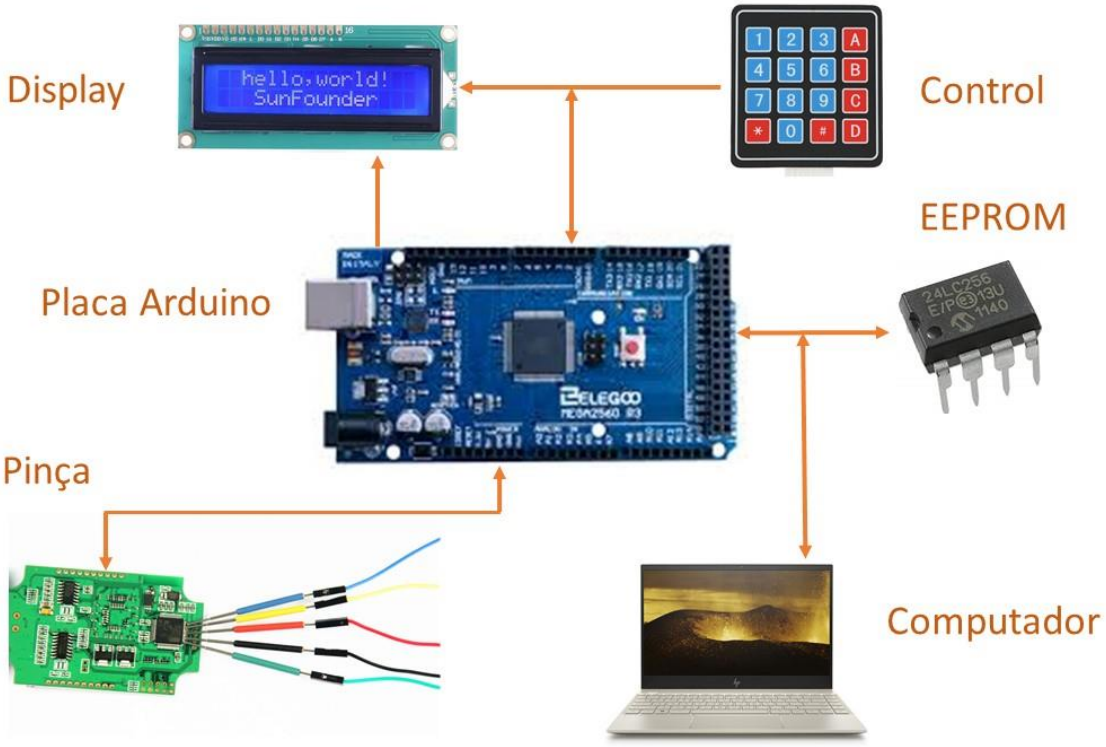


Figura 1 Estructura

3.2.2. Modes de funcionament

Abans d'escollir el mode, el correcte ús del testejador dicta tres passos bàsics, el primer és l'encesa del dispositiu un cop ha sigut connectat al corrent. A continuació es procedeix a fer la pertinent connexió de la pinça al circuit i finalment es selecciona el mode desitjat.

Existeixen quatre modes possibles amb requisits i resultats diferents, els quals són escollits a partir del menú principal. Ordenats segons l'aparició en el menú són:

1. Identificació de l'IC, "Identify IC": és el mode principal del testejador, es connecta la pinça al circuit a testejar el qual ha de trobar-se inactiu. Quan es selecciona el mode en el menú el processador comença a recórrer l'EEPROM llegint els valors. Quan troba el títol d'un circuit integrat el mostra sobre la LCD indicant el nombre de pins. Depenent de la taula de veritat emmagatzemada a l'EEPROM i vinculada al circuit integrat, la placa d'Arduino aplicarà els Inputs, impulsos d'entrada al circuit, i llegirà les sortides Outputs. Aquestes sortides seran contrastades amb els valors de la mateixa taula de veritat. En cas que concordin, el nom del circuit integrat actual apareixerà en pantalla indicat com el xip correcte.

2. Anàlisi d'IC, "Analyze_IC": en aquesta variant l'IC a testejar s'ha de trobar en mode operatiu, un cop connectada la pinça a l'IC i accionat el mode, apareixeran l'estat de cada pin en pantalla. Donat que tots els pins estan funcionant com a input, els valors obtinguts només poden discernir entre 0 o 1, no existeix H o L, llavors no es pot conèixer si un pin actua com a entrada o sortida. No obstant. si un pin no està connectat o la pinça no rep senyal, el processador retornarà el valor de X.

3. Veure data, "View data": seleccionant aquest submenú l'usuari pot llegir totes les taules de veritat emmagatzemades en la EEPROM, classificades segons IC.

4. Reemplaçar IC, "Replace IC": aquest mode permet que el testejador imiti el comportament d'un IC qualsevol, sempre digital, segons una única línia de la seva taula de veritat aconseguint la substitució parcial de l'IC. En escollir el nombre de pins desitjat es procedeix a introduir en pantalla tots els valors que es desitja aplicar a través de la pinça. No obstant, no existirà diferència entre 1,V,H i 0,G,L separadament ja que tots els valors són inputs que equivalen a 1 o 0.

3.3. HARDWARE

En cas que el lector no estigui familiaritzat amb el terme que capitula aquest apartat, s'entén Hardware com la implementació física i material d'un dispositiu electrònic, contrari al concepte de Software el qual engloba tota la programació informàtica i "fictícia" del producte.

Com s'ha explicat a l'apartat anterior, el bloc funcional més important és el microcontrolador d'Arduino. Aquest en qüestió és el microcontrolador Arduino Mega 2560. La raó d'utilitzar aquesta placa i no una altra d'Arduino, és que disposa de nombrosos pins per connectar el cablejat que comunica tots els blocs funcionals, en concret disposa de 54 pins digitals i 16 pins analògics. Té dues entrades d'alimentació, una per connectar-se mitjançant un adaptador AC-DC directament a l'endoll domèstic i l'altra de format USB que a la vegada serveix per carregar el codi des de l'ordinador. Per alimentar els components electrònics a connectar al microcontrolador, aquest disposa de nombrosos pins amb sortides de 5V i 0V o a partir d'ara *ground*.

L'Arduino Mega 2560 funciona amb el microprocessador ATmega2560 encarregat de fer tots els càlculs i operacions. Aquest xip disposa d'una memòria de 256K on s'emmagatzema tot el codi escrit per ordinador.

Per facilitar les connexions al microcontrolador d'Arduino i disminuir el cablejat s'habiliten punts o línies de connexió comuns de *ground* i alimentació que després es connecten únicament a un pin de la placa. Com s'ha comentat, el punt d'alimentació d'Arduino es troba a 5V suficient per suportar tots els components a connectar i insuficient per sobrepassar el

límit de voltatge de qualsevol.

L'explicació detallada dels components electrònics que conformen el hardware del dispositiu s'exposarà en el mateix ordre expressat en l'apartat *3.2.1 Estructura,* obviant el computador.

3.3.1. EEPROM 24LC256

La informació exposada a continuació ha sigut extreta de la *datasheet* del component 24LC256. Es troba adjuntada a la bibliografia per a consultes amb més detall [6].

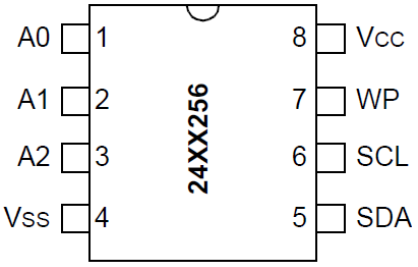


Figura 2 EEPROM

La EEPROM 24LC256 s'alimenta amb un rang de 2.5V-5.5V mitjançant el pin 8, Vcc.

Pin 4, Vss es connectarà a terra, *ground*.

Els pins 5 i 6 fan referència a la transmissió de dades. El 5, SDA - Serial Data s'encarrega de l'entrada i sortida de dades, mentre que el pin 6, SCL – Serial Clock, dictamina els instants o períodes de temps en els quals es realitza la transmissió.

La lectura de la memòria sempre es troba habilitada, no obstant, el pin 7, WP que representa Write-Protect haurà d'estar connectat a *ground* per habilitar l'escriptura sobre la memòria.

Finalment pins 1,2,3 d'entrada analògica com es pot observar a la Figura 2 s'utilitzen en operacions de múltiples dispositius. En aquest cas al disposar d'una única memòria connectarem aquests pins directament a *ground* per obtenir el valor 0 als tres pins.

S'ha pogut observar que la comunicació de dades es dona en sèrie. En aquest tipus de transmissió la informació s'alinea en una cua i es transmet bit per bit, fent ús d'un únic canal. La comunicació en paral·lel és molt més ràpida ja que els bits viatgen simultàniament però requereixen molt més cablejat.

Així doncs, pins 1,2,3,4 i 7 seran connectats a la línia de *ground*, pin 8 a la de 5V i SDA i SCL als pins 20 i 21 respectivament de la placa Mega 2560.

3.3.2. LCD 1602 Module

La informació exposada a continuació ha sigut extreta de la *datasheet* del component LCD 1602. Es troba adjuntada a la bibliografia per a consultes amb més detall [7].

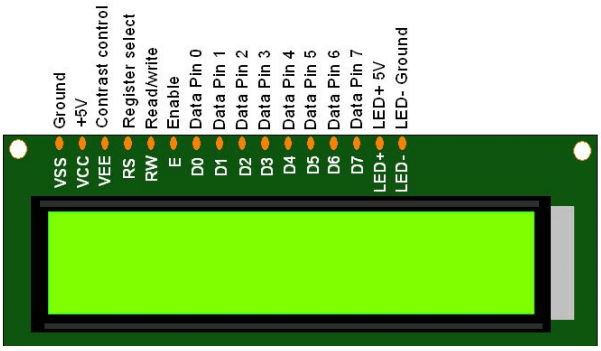


Figura 3. LCD

La LCD – "Liquid crystal display", pantalla de cristall líquid, disposa com el seu codi indica de 16x2 32 caselles d'escriptura. Cada casella conforma una matriu d'escriptura 8 files per 5 columnes on cada posició representa un bit 0 si és invisible 1 si està escrit.

Tenint en compte la *Figura 3* l'enumeració dels pins serà ascendent d'esquerra a dreta.

Pel que fa la alimentació, els pins 1 i 16 es connecten a *ground*, els pins 2, Vcc i 15, LED a 5V.

El pin 3 fa referència al contrast de la pantalla entre el fons i els caràcters a escriure. Aquest pin també es connecta a 5V però a través d'un potenciòmetre. El potenciòmetre és una resistència variable. El que s'utilitza en aquest projecte és el de rotació de 10K. Disposa d'una pista plena de material resistiu, carbó, per on ha de circular el corrent, depenent del control extern es pot augmentar o disminuir el recorregut d'aquesta pista. Es pot observar l'esquema a la Figura 4. El terminal B, la sortida variable, es connecta al pin 3, Contrast Control, el terminal A, a 5V i el terminal C a *ground*.

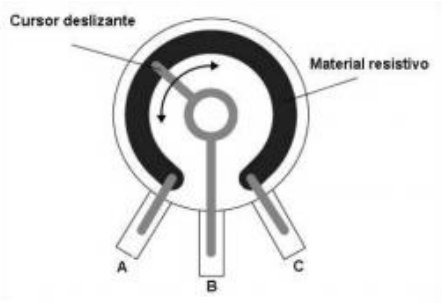


Figura 4 Potenciòmetre

El pin 5 R/W que fa referència al mode de lectura d'escriptura es connectarà a *ground* per escriure.

Els pins 4,6,11,12,13 i 14 es connectaran als pins lògics d'Arduino per controlar i enviar la informació a mostrar.

3.3.3. Control

El control i interacció de l'usuari amb el testejador es durà a terme emprant interruptors de botó. Aquests polsadors permeten el pas de corrent entre terminals quan es prem la peça superior mòbil, immediatament després d'alliberar la pulsació el corrent es talla. Per generar

un interruptor constantment actiu cal aplicar tensió al terminal d'entrada. El terminal de sortida primer es connecta a terra mitjançant una resistència per garantir el corrent, en segon lloc es connecta a un pin lògic d'Arduino quedant aquesta connexió en paral·lel amb la resistència. Cada cop que es premi l'interruptor, s'enviarà corrent al pin lògic d'Arduino que traduït pel programa activarà o desactivarà funcions del codi. S'implementaran 5 botons: 4 fletxes de direccions i una de selecció. Per a facilitar l'ús del control per part de l'usuari, cada botó disposarà d'un LED propi que s'il·luminarà indicant que si es prem l'interruptor, aquest generarà algun efecte sobre el testejador. Cada LED anirà associat a una resistència en sèrie per a disminuir la lluminositat i protegir-los de la sobretensió.

A més s'introduiria un últim interruptor d'encesa posicionat entre l'alimentació i el testejador.

3.3.4. Pinça electrònica

La pinça electrònica és el pont d'unió entre el microcontrolador i l'IC a testejar. Totes les connexions entre els components exposats són d'agradable senzillesa donat que les peces disposen de terminals adaptats per introduir o acoblar el cablejat. No obstant, en intentar unir el IC-Tester a un xip determinat nombroses complicacions apareixen.

La primera complicació està totalment relacionada amb l'atribut de versatilitat i maniobrabilitat del testejador ja que aquest està dissenyat per analitzar circuits integrats sense desacoblar-los del circuit que completen, evitant dessoldar o reparacions complexes. L'estudi realitzat sobre el xip s'ha de fer de manera puntual, estable, eficaç i sense deixar cap tipus de rastre. Així doncs la unió al xip s'haurà de fer per la cara superior acoblant la pinça a la reduïda superfície exposada de l'encapsulat.

D'aquesta unió neix la tercera simplificació del projecte, referent també als tipus de circuits integrats. Només es podran testejar xips que s'hagin fabricat en l'encapsulat DIP, "dual in-line package", empaquetament de doble línia i de 19 mm de llargada per 5 mm d'amplada, aproximadament. S'ha tendit per aquest tipus de empaquetament per la senzillesa de la seva estructura rectangular amb l'objectiu d'obtenir un acoblament estable amb la pinça.

La segona problemàtica d'unió recau en la complexitat de treballar amb les tecnologies que l'autor disposa sobre dimensions reduïdes, és per aquesta dificultat de connexions mil·limètriques que és necessària una simplificació addicional. Només es testejaran circuits integrats de 14 i 16 pins. La raó per la qual no s'amplia el projecte a xips amb més pins és per la necessitat de crear una nova pinça, enfocant el projecte només a 14 i 16 pins es pot dissenyar una pinça adaptable a ambdós casos.

Finalment per al correcte funcionament dels modes exposats a l'apartat *3.2.2 Modes de*

funcionament seran necessàries dues pinces, una connectada als pins digitals d'Arduino desenvoluparà els modes referents a identificar l'IC i a reemplaçar l'IC, la segona pinça, de caire analògic, s'implementarà únicament amb el mode analitzar IC. La raó d'aquesta distinció és que en el mode d'anàlisi de l'IC, aquest es troba en funcionament i els corrents que transcorren els seus pins han de ser llegits en mode analògic per ser correctament diferenciats.

3.4. SOFTWARE

3.4.1. Introducció al Codi d'Arduino

Un cop implementat el hardware de tot el dispositiu es requereix d'un codi que controli la placa base i consolidi totes les connexions físicament realitzades.

El codi ha sigut redactat utilitzant el programari de codi obert d'Arduino. El codi obert, *open source* en anglès, és un model de desenvolupament de software basat en la col·laboració oberta entre els usuaris del programa. En concret, pel que fa al codi exposat a continuació moltes funcions ja creades han sigut obtingudes de tutorials i exemples oferts per Arduino. Per altra banda el codi elaborat en aquest projecte serà pujat a la base de dades d'Arduino per a què qualsevol usuari interessat pugui disposar-ne.

El programari IDE suporta el llenguatge de programació C++. Aquest llenguatge, com molts altres, és composat per un seguit d'estructures. A continuació es llistaran les més rellevants.

- Variables: ocupen la major part de la memòria del codi, quan s'inicialitza una variable se li atribueix un valor, coherent al tipus de variable inicialitzat. Un cop és creada, pot ser copiada, modificada o cridada tants cops com es desitgi. Existeixen nombrosos tipus de variables, les que utilitzarem en el codi són:
    - Int: integradors, nombres enters que oscil·len entre -32768 i 32767.
    - Char: caràcter, engloba tots els possibles símbols, lletres i números del teclat, traduïts mitjançant el sistema ASCII. Com es pot observar a la taula de la bibliografia [8] les variables Char es mouen dins del rang de 0 a 128.
    - String: cadena, molt semblant a la variable de tipus Char donat que engloba el mateix rang de valors, no obstant fa referència a una cadena de caràcters Char, no només un valor.

- Bool: variable que pren els valors "true" o "false", vertader o fals, útils per verificar les estructures de condicions explicades més endavant.

- Funcions: és l'estructura més genèrica. Al definir una funció, aquesta disposa d'un seguit de variables i d'accions que s'executaran una darrera l'altra cada cop que la funció sigui cridada en algun apartat del codi.

- Iteracions: les estructures d'iteracions permeten realitzar una mateixa operació sobre variables que poden ser modificades. El nombre d'iteracions és infinit fins que la condició establerta per l'estructura és satisfà.

- Condicions: estructura prèviament esmentada, les condicions permeten versatilitat al codi donat que es creen diversos camins o opcions pels qual transcórrer. Quan la condició inicial es satisfà el text contingut dins de l'estructura es executat. Si no es dona la validació de la condició, el text és obviat.

És la combinació d'aquestes estructures la que dona una àmplia llibertat als llenguatges de programació podent realitzar tasques molt complexes.

3.4.2. Disseny del codi

Quan es tracta de dissenyar un codi llarg i complex el més important és entendre els objectius principals i secundaris i definir les funcions amb coherència, les demés estructures són conseqüència del objectiu de la funció. Pel que fa al codi del testejador existeixen dos tipus de funcions a implementar exposades a continuació.

3.4.2.1. Funcions complementàries

La naturalesa complementària d'aquestes funcions recau en la tasca que desenvolupen, pretenen donar suport a les funcions més generals. Realitzen un treball concret sense possibilitat de ser modificat mitjançant els paràmetres de control. A més, aquestes funcions requereixen ser cridades amb un seguit d'informació com a variable d'entrada sobre la qual treballen. En aquest codi es disposa de tres funcions complementàries.

Write_Data (int adress, char val): funció encarregada d'ordenar i gravar informació a la memòria programable, requereix de dues variables d'entrada, la primera fa referència a l'integrador que indica la posició d'adreça on començar a escriure, la segona entrada són els valors de caràcters a gravar dins de les adreces. Simultàniament Write_Data ordena la informació, es va cridant la subfunció **Write_EEPROM** la qual estableix la transmissió amb la memòria i explícitament grava la informació segons les variables d'entrada de Write_Data.

Com es pot observar a codi en l'annex A.6 Funció Write_Data a cada valor de codi gravat l'adreça s'incrementa en una unitat avançant per la memòria. Write_Data ha sigut l'única funció escrita en una pàgina de projecte diferent, així l'escriptura de la EEPROM s'ha de fer prèvia i separadament a l'ús del testejador.

Read_EEPROM (int adress): funció encarregada de llegir els valors emmagatzemats a la memòria. Observem que només es requereix una entrada referent a l'adreça d'inici de lectura. L'increment del valor d'aquesta entrada haurà de redactar-se en la funció que ha cridat a Read_EEPROM .

Display_Lcd (String text, int line): té com objectiu la visualització de text sobre la pantalla LCD, el text és enviat en format String, i l'entrada *line* fa referència al número línia on es vol escriure el text. Dins del text trobem diverses variants d'aquesta funció necessàries per visualitzar text de característiques particulars, com fletxes de direcció amb la funció Display_Lcd_Arrows.

Pel que fa al control mitjançant els polsadors i les entrades d'impulsos digitals, no s'ha definit cap funció en si mateixa, sinó que les condicions de control i la seva possible satisfacció van apareixent al llarg del codi a cada funció.

3.4.2.2. Funcions principals

A diferència de les complementàries, les funcions principals no donen suport a cap altra funció ni es subordinen a la freqüent crida per aportar informació o modificar variables. De fet el resultat d'aquestes funcions es resumeix en enviar mitjançant Display_Lcd text a la pantalla o impulsos a través de la pinça. Cada funció principal fa referència a un dels modes ja presentats al apartat 3.2.2 Modes de funcionament

Indentify_IC () **Analyze_IC(numpins)** **View_Data()** **Replace_IC**

Existeix una última funció a esmentar fora de la classificació prèvia entre complementàries i principals. Aquesta funció és **Menu()**, és la primera funció en ser executada que combinada amb els botons de control permet seleccionar el mode de funcionament desitjat.

Les cinc funcions es poden trobar a ANNEX A: CODI.

4. CONSTRUCCIÓ

El procés de construcció de l'IC Tester consta de dues fases. La primera, de prototipatge, té com objectiu verificar la funcionalitat del disseny escollit, donar garanties d'èxit del producte. Com a totes les verificacions és molt probable que s'hagi de modificar el disseny i tornar a testejar-lo de forma iterativa. Un cop el prototip compleix les especificacions es podrà prosseguir amb la segona fase, la construcció definitiva. S'empaquetaran tots els components electrònics en una carcassa còmoda i segura per a l'usuari, garantint la facilitat de control i lectura i protegint els components electrònics interiors.

4.1. Prototipatge

L'element bàsic en el prototipatge de circuits electrònics és la *Protoboard* o placa de proves, composada per un conjunt de blocs connectats per materials conductius creant una sèrie de línies en paral·lel. Cada línia disposa de múltiples orificis per introduir els terminals dels components electrònics sense la necessitat d'aplicar soldadura. Degut a la naturalesa reutilitzable i reconfigurable de la *Protoboard* esdevé una eina molt útil per a la verificació i modificació del disseny en la fase de prototipatge.

A l'ANNEX B: PROTOTIPATGE es pot observar amb precisió el prototip en conjunt i de cada bloc funcional sobre la *Protoboard*, a més de l'esquemàtic de totes les connexions. Les resistències utilitzades són del valor nominal de 1kΩ (1000 Ohms) per il·luminar els LEDs de 3mm amb suficient intensitat i protegir-los del perill a sobrecàrrega.

4.2. Construcció definitiva

Finalitzades totes les iteracions disseny-prototip s'encara la tasca final de construcció. Aquest procés es pot dividir en dues fases ben diferenciades: la fabricació de la carcassa i l'assemblatge intern de tots els components.

4.2.1. Fabricació de la carcassa

Com s'ha comentat, per protegir i garantir la durabilitat de totes les connexions s'empaquetaran tots els components electrònics dins d'una caixa o carcassa. Únicament restaran visibles la pantalla els botons de control i les connexions d'alimentació i de transmissió de dades de la placa d'Arduino.

Donada la complexitat de les geometries a emmagatzemar dins la carcassa, s'ha optat per elaborar-la mitjançant la impressió 3D. El programari emprat per al disseny ha estat

SolidWorks. Per aportar comoditat al futur assemblatge, tot el conjunt ha estat dividit en set peces diferents: dues pinces, dues tapes, el caixetí, una paret de tancada i la clau d'obertura de la carcassa. Tots els plànols es poden consultar a ANNEX C: PLÀNOLS CARCASSA.

Com s'ha comentat a l'apartat de hardware, es requereixen dues pinces: analògica i digital. No obstant, pel que fa a la fabricació aquestes són idèntiques, les connexions del cablejat s'apliquen per la cara superior de la peça. A la cara inferior resideix la ranura que contactarà la cara superior del xip. Observant els plànols es pot percebre un lleuger aplanament de les cantonades de la ranura. És en aquest contacte per deformació plàstica on es fonamenta la subjecció de la pinça al circuit integrat.

El sòlid Caixetí està composat per quatre parets en forma de prisma. A la cara inferior s'alcen unes petites geometries en forma de con per introduir-se a través dels orificis de la placa d'Arduino i estabilitzar-la. Les parets laterals, quasi simètriques, disposen de tres suports. Sobre l'inferior es posicionaran seccions de la *Protoboard* per realitzar les connexions interiors. En el suport mitjà es dipositarà la primera tapa la qual sustenta la LCD, els interruptors i els LEDs. Sobre el suport superior recau l'última tapa que juntament amb la mitjana conclouen l'empaquetament dels tres components electrònics previs. La paret restant, mitjançant tres perforacions, permet la connexió de dades i d'alimentació, a la vegada que l'assentament de l'interruptor d'encesa.

L'asimetria de les parets laterals es fonamenta en què la paret de la dreta (observant C.1 Plànol de peça Caixetí) disposa d'una ranura on introduir la clau d'obertura per a què les parets de la carcassa quedin fixades. Aquesta clau formada per l'extrusió d'un perfil circular amb l'acoblament d'un triangular pot ser introduïda i retirada a través de la paret de tancada. Un cop introduïda, la clau ha de ser rotada 180º fins a coincidir amb la ranura de la paret lateral. Degut a les geometries del nou contacte, la paret de tancada queda limitada del seu desplaçament vertical bloquejant l'obertura del conjunt. La clau d'obertura disposa d'una ranura a la cara exterior per facilitar la rotació. Amb aquest senzill sistema mecànic s'aconsegueix el tancament de l'estructura sense la necessitat de pega o cargols. Així l'usuari pot obrir la carcassa i modificar l'interior fàcilment sempre que es vulgui.

4.2.2. Assemblatge

El procés d'assemblatge de tots els components dins del caixetí i el posterior tancament de la carcassa és una tasca que requereix summa paciència i concentració degut a l'alta compactació de components en el reduït espai on operar. La dificultat, però, es redueix al seguir la guia exposada a continuació; tota la informació referent a l'assemblatge es pot consultar en aquesta.

5. GUIA DE CONSTRUCCIÓ

Seguint el propòsit d'una major divulgació, s'ha decidit elaborar la guia de construcció en llengua anglesa. La guia consta d'una breu introducció i 16 passos o *steps* a seguir. Es pot consultar a l'ANNEX D: GUIA DE CONSTRUCCIÓ.

La introducció, acompanyada d'un petit text, serveix per aclarir a l'usuari la càrrega de treball que suposa la construcció, indicant la dificultat de fabricació, el nombre de passos i el temps aproximat de fabricació.

El primer *step* mostra tots els materials o components que s'utilitzaran en el procés. Cada *step* següent està format per un encapçalament, una imatge descriptiva i unes curtes indicacions de les accions a realitzar en el respectiu pas.

La guia de construcció no exposa com ha sigut redactat el codi d'Arduino o el modelat de les peces mitjançant SolidWorks. No obstant, en la divulgació del document s'adjuntaran tots els arxius referents a la realització d'aquest projecte, tots menys la memòria mateixa.

Els *steps* o passos de la guia són:

0. Introducció.
1. Llista de materials.
2. Col·locar l'Arduino.
3. Col·locar l'interruptor d'encesa.
4. Col·locar *Protoboard*
5. Col·locar i alimentar l'EEPROM.
6. Col·locar les resistències i el potenciòmetre.
7. Col·locar interruptores i LEDs.
8. Col·locar la LCD i els botons.
9. Alineació de tapes.
10. Connexions.
11. Cablejat de la pinça.
12. Connexió de la pinça
13. Introduir les tapes i parets.
14. Introduir la clau
15. Acoblar la pinça.
16. Fi.

6. Planificació i pressupost

Un cop definits els objectius que han guiat aquest projecte es va decidir seguir el següent calendari que mostra les fases prèvies i posteriors, amb les seves respectives tasques.

	Tasca	Fase
Juny	Electrònica i components	Recerca
Juliol	Reparació en la electrònica	
Setembre	Viabilitat de l'IC Tester	
Octubre	Programació Informàtica	Disseny
Novembre	Circuit electrònic	
	Prototipatge	
Desembre	Modelatge de la Carcassa	
Gener	Construcció definitiva	Construcció
Gener	Guia de construcció	
Desembre, Gener	Redacció memòria	Documentació
Febrer	Divulgació de la guia	

Taula 1 Calendari

En l'àmbit de pressupost, dos tipus de costos han sigut presents, un és el cost de l'autor com a enginyer per totes les hores de treball dedicades.

El segon, és el cost dels materials utilitzats per a la fabricació del IC Tester, no s'han incorporat les possibles despeses relacionades amb programari ja que Arduino es *open source* i la llicència de SolidWorks implementada és la estudiantil, de cost zero per a l'autor. El balanç es

pot consultar a la següent taula:

Servei o Producte	Quantitat	Cost unitari	Total
Enginyer	696h	15*	10,440.00€
Arduino Mega 2560 Kit	1	52	52.00 €
Protoboard	1	5	5.00 €
Circuits Integrats	8	0.4	3.20 €
Impressió 3D	30h	0.85	25,5 €
TOTAL			**10525,7 €**

Taula 2 Pressupost

**Impostos ja aplicats*

És important remarcar que els costos associats a la quantitat d'hores aplicades per l'enginyer en el projecte només es donen en aquesta primer lot. Per tornar a fabricar l'IC Tester només caldria invertir el cost associat als materials una quantitat aproximada de 4 hores per a l'assemblatge.

Conclusions

Emprant un enfocament global, a l'inici del projecte, ignorant era l'autor en creure que només treballaria en el camp de l'electrònica. Aquest document llavors, es pot considerar un bon fruit de l'escola que exemplifica el tarannà del Grau en Enginyeria en Tecnologies Industrials, un precís equilibri entre les diverses disciplines que sostenen l'Enginyeria. Projecte que ha requerit coneixements previs i aprenentatge simultani de programació, modelatge, fabricació i evidentment, electrònica.

Aplicant una visió més concreta sobre el projecte, l'autor considera que s'han assolit de manera satisfactòria els objectius plantejats a l'inici del document: Dissenyar-Construir-Divulgar. No obstant això, diverses simplificacions han estat necessàriament implementades, atemptant contra la resolució de l'objectiu principal, l'elaboració de l'IC Tester. De totes maneres, l'existència d'aquestes simplificacions no ha afectat l'essència del producte del projecte, només ha reduït el seu abast d'aplicació. Tot i que involuntàriament, es podria dir que s'ha optat per qualitat envers quantitat.

Per atribuir una mica d'objectivitat sobre aquestes conclusions tan personals, el més imponent indicatiu per concloure sobre la correcta realització del projecte o document, recau en la futura utilització de la guia de construcció i de l'IC Tester mateix, per altres usuaris o usuàries.

Agraïments

Quatre són els agraïments que desitjo mostrar.

El primer, és dedicat a tot el bon professorat que mitjançant la docència com a campana ha despertat la meva curiositat més interna sobre les profunditats de l'enginyeria i totes les ciències relacionades que es compliquen per entendre la realitat.

A continuació, agrair a la gran Cristina Lampón Diestre en el seu debut com a tutora del TFG, tot un exemple d'equilibri entre paciència i visió crítica.

Moltes gràcies a l'Arnau Gol membre de *Chuscateds Aproved Design* pel suport en el modelatge i impressió 3D.

Finalment, cap paraula hauria tingut cabuda en aquest document sense l'incondicional esforç i ajuda d'Arduino.CC que permet a tants com jo, somiar en el món de l'electrònica.

Bibliografia

Referències bibliogràfiques

[1] AMERICAN PHYSICAL SOCIETY*, This Month in Physics History October 1897: The Discovery of the Electron*
https://www.aps.org/publications/apsnews/200010/history.cfm

[2] INSTRUCTABLES - https://www.instructables.com/

[3] WIKIPEDIA, https://en.wikipedia.org/wiki/List_of_7400-series_integrated_circuits

[4] WIKIPEDIA , https://en.wikipedia.org/wiki/List_of_4000-series_integrated_circuits

[5] JORBI,*TEST_16_FULL_0004.DAT*
https://cdn.instructables.com/ORIG/F5K/PGEU/IK2UZ059/F5KPGEUIK2UZ059.dat

[6] MICROCHIP TECHNOLOGY INC, *24AA256/24LC256/24FC256 CMOS SERIAL EEPROM ,* http://ww1.microchip.com/downloads/en/devicedoc/21203m.pdf

[7] TINSHARP, *TC1602A-01T,* LCD
https://cdn-shop.adafruit.com/datasheets/TC1602A-01T.pdf

[8] ASCIITABLE.XYZ, *ASCII TABLE*

https://www.asciitable.xyz/

[9] ARDUINO - https://www.arduino.cc/

ANNEXES

ANNEX A: CODI

o A.1 Funció Menú

```
void Menu(){

  int select = 0;    //pin25
  int up = 0;        //pin26
  int down = 1;      //pin27
  int left = 0;       //pin28
  int right = 0;     //pin29

  digitalWrite(35,HIGH);
  digitalWrite(36,LOW);
  digitalWrite(37,HIGH);
  digitalWrite(38,LOW);
  digitalWrite(39,LOW);
  pinMode(22,OUTPUT);
  digitalWrite(22,HIGH);

  int menu_number = 0;
  while(select !=1 ){

    if(down==1){
      menu_number = menu_number +1;
      if(menu_number==1){
        Display_Lcd_Arrow("1. Indentify IC","",1,1);
        Serial.println("1. Indentify IC");
        delay(500);}

      if(menu_number==2){
        Display_Lcd_Arrow("2. Analyze_IC","",1,1);
        Serial.println("2. Analyze_IC");
        delay(500);}

      if(menu_number==3){
        Display_Lcd_Arrow("3. View data","",1,1);
        Serial.println("3. View data");
        delay(500);}

      if(menu_number==4){
        Display_Lcd_Arrow("4. Replace IC","",1,1);
        Serial.println("4. Replace IC");
        menu_number = 0;
        delay(500);}
    }
    down = digitalRead(27);
    select = digitalRead(25);
  }
  digitalWrite(35,LOW);                    //all of them low
```

```
digitalWrite(36,LOW);
digitalWrite(37,LOW);

if(menu_number == 1){
  Display_Lcd_Arrow("Start the search","",0,0);
  delay(2000);
  digitalWrite(36,HIGH);
  Search_for_IC();}

if(menu_number == 2){
  Display_Lcd_Arrow("Select number of pins","",0,0);
  delay(2000);
  int numpins = 0;
  while(true){
    down = 0;
    digitalWrite(35,HIGH);
    digitalWrite(37,HIGH);
    while(down == 0){
      Display_Lcd_Arrow("14","16",4,0);
      delay(300);
      Display_Lcd_Arrow("14","16",0,0);
      delay(300);
      numpins = 14;
      down = digitalRead(27);
      select = digitalRead(25);
      Serial.println("while 14");
      if(select == 1){break;}
    }
    if(select == 1){break;}
    select = 0;
    down = 0;
    while(down == 0){
      Display_Lcd_Arrow("14","16",4,1);
      delay(300);
      Display_Lcd_Arrow("14","16",0,0);
      delay(300);
      numpins = 16;
      down = digitalRead(27);
      select = digitalRead(25);
      Serial.println("while 16");
      if(select == 1){break;}
    }
    if(select == 1){break;}
  }
  Display_Lcd_Arrow("Let's Analyze_IC ","",0,0);
  delay(2000);
  Analyze_IC(numpins);}

if(menu_number == 3){
  delay(2000);
  digitalWrite(39,HIGH);
  View_Data();}

if(menu_number == 0){
  Display_Lcd("Start the replacement",0);
  delay(2000);
  Replace_IC();} }
```

o A.2 Funció Identificació de l'IC

```
void Indentify_IC (){

 int adress = 0;
 char val = Read_Eeprom(adress);
 char numpins[3]={'\0'};

 int up = 0;
 while(up==0){
 while(val != '&'){                    // while not finished
  up = digitalRead(26);
  if(up==1){Menu();}
  if(val == '$'){
    char Name[5]={'\0'};
    int index = 0;

    while(val != ' '){
      adress = adress + 1;
      Name[index] = (char)(Read_Eeprom(adress));
      index = index + 1;
      val=Read_Eeprom(adress);}
    Serial.println("Next IC:");
    Serial.print(Name);
    Serial.println(".");  //
    numpins[0]=(char)(Read_Eeprom(adress+1));
    numpins[1]=(char)(Read_Eeprom(adress+2));
    Serial.print("n°pins: ");
    Serial.println(numpins);
    Display_LcdTwowords("Checking IC: ",Name);
    delay(1000);

    if(numpins[1] == '4'){
      Serial.println("Start Check_pins_14: ");
      if(Check_pins_14(adress) == 1){                    //returns 0 o 1
        Serial.print("The IC ");
        Serial.print(Name);
        Serial.println(" is the CORRECTIC");
        Display_Lcd_Arrow("   Correct Ic",Name,0,0);
        break;}
        //Some break of the while != &
      else{
        Serial.print("The IC ");
        Serial.print(Name);
        Serial.println(" was not the correctIC");}
      Serial.print("Stop Check_pins_14 at adress: ");
      Serial.println(adress);
      Serial.println("");
      Serial.println("");
      Serial.println("");}


    if(numpins[1] == '6'){
      Serial.println("Start Check_pins_16: ");
      if(Check_pins_16(adress) == 1){                    //returns 0 o 1
        Serial.print("The IC ");
```

```
        Serial.print(Name);
        Serial.println(" is the CORRECTIC");
        Display_LcdTwowords("Correct Ic:",Name);
        break;}
        //Some break of the while != &
      else{
        Serial.print("The IC ");
        Serial.print(Name);
        Serial.println(" was not the correctIC");}
      Serial.print("Stop Check_pins_16 at adress: ");
      Serial.println(adress);
      Serial.println("");
      Serial.println("");
      Serial.println("");}
      }

   adress = adress + 1;
   val = Read_Eeprom(adress);}

 Serial.println("Data finished");
 int up = 0;
 up = digitalRead(26);}

 Menu();}

bool Check_pins_14(int adress){
  int up = 0;
  while(up==0){
  adress = adress + 4;            //15
  bool correctIc = true;
  char check_val = Read_Eeprom(adress);
  Serial.println("checking14...");

  while(((int)check_val != 36)&&((int)check_val != 38)){
    up = digitalRead(26);
    if(up==1){Menu();}
    int inadress = adress;                  //15        30
    int outadress = adress;                 //15
    int clockadress = adress;
    char inval = (char)Read_Eeprom(inadress);     //L
    char outval = (char)Read_Eeprom(outadress);   //L
    char clockval = (char)Read_Eeprom(clockadress);
    char posval[10] = {' ','0','1','L','H','G','V','X','C'};
    //(int)posval[10]={ 32,48, 49, 76, 72, 71, 86, 88, 67}
    int inpin = 2;
    int outpin = 2;
    int clockpin = 2;

    pinMode(8, OUTPUT);       // Input G and V all the time until &-NewIC
    digitalWrite(8,LOW);
    pinMode(15, OUTPUT);
    digitalWrite(15,HIGH);

    while((int)inval != 32){                        // INPUTS 1 0 to IC
      Serial.println(" ");

      if((int)inval == 48){       // 0
        pinMode(inpin, OUTPUT);
```

```
    digitalWrite(inpin,LOW);
    delay(100);}

  if((int)inval==49){       // 1
    pinMode(inpin, OUTPUT);
    digitalWrite(inpin,HIGH);
    delay(100);}

  Serial.print("");
  Serial.print("Pin ");
  Serial.print(inpin);
  Serial.print(" sets: " );
  Serial.println(inval);

  up = digitalRead(26);
  if(up==1){Menu();}

  inpin = inpin + 1;
  inadress = inadress + 1;
  inval = Read_Eeprom(inadress);}

while((int)clockval != 32){ //Clock once all inputs have been done
  Serial.println("");
  if((int)clockval == 67){
    Serial.println("Triggering Clock");
    pinMode(clockpin, OUTPUT);
    digitalWrite(clockpin,LOW);                    // set it  LOW
    delay(100);
    digitalWrite(clockpin,HIGH);                   // set it  HIGH
    delay(100);}

  up = digitalRead(26);
  if(up==1){Menu();}

  clockpin = clockpin + 1;
  clockadress = clockadress + 1;
  clockval = Read_Eeprom(clockadress);}

while((int)outval !=32){                         // OUTPUTS from IC
  Serial.println(" ");
  if(((int)outval == 76) || ((int)outval == 72)){
    pinMode(outpin, INPUT_PULLUP);
    int response = digitalRead(outpin);
    Serial.print("Response at outpin ");
    Serial.print(outpin);
    Serial.print(" has been: ");
    Serial.println(response);
    Serial.print("And it should be: ");
    Serial.println(outval);
    //convert response to L H
    int convertoutval = 1;
    if((int)outval == 76){
      convertoutval = 0;}
    if((int)outval == 72){
      convertoutval = 1;}
    Serial.print("response: ");
    Serial.println(response);
    Serial.print("convertoutval: ");
```

```
        Serial.println(convertoutval);

        if(response != convertoutval){  //Response = IC ; Outval = Eeprom
          correctIc = false;
          Serial.println("FALSE at: ");
          Serial.println(outadress);
          break;}
        }
      up = digitalRead(26);
      if(up==1){Menu();}

      outpin = outpin + 1;
      outadress = outadress +1;
      outval = Read_Eeprom(outadress);}

    if (correctIc == false){
      break;}
    adress = inadress + 1;                  //29+1=30
    check_val = Read_Eeprom(adress);
    Serial.println(" ");
    Serial.println(" ");
    Serial.println(adress);
    Serial.println(check_val = Read_Eeprom(adress));
    Serial.println(" ");
    Serial.println(" ");
    Serial.println(" ");}
  Serial.println("End of Checkpins14 and pin reset");
  int pinreset = 2;
  while(pinreset<=15){
    digitalWrite(pinreset,LOW);
    pinreset =  pinreset +1;}
  return correctIc;}
}

bool Check_pins_16(int adress){
  adress = adress + 4;
  bool correctIc = true;
  char check_val = Read_Eeprom(adress);
  Serial.println("checking16...");
  int up = 0;

  while(((int)check_val != 36)&&((int)check_val != 38)){
    up = digitalRead(26);
    if(up==1){Menu();}
    int inadress = adress;                  //15        30
    int outadress = adress;                 //15
    int clockadress = adress;
    char inval = (char)Read_Eeprom(inadress);     //L
    char outval = (char)Read_Eeprom(outadress);   //L
    char clockval = (char)Read_Eeprom(clockadress);
    char posval[10] = {' ','0','1','L','H','G','V','X','C'};
    //(int)posval[10]={ 32,48, 49, 76, 72, 71, 86, 88, 67}
    int inpin = 2;
    int outpin = 2;
    int clockpin = 2;

    pinMode(9, OUTPUT);                                 // Input G and V
all the time until &-NewIC
```

```
  digitalWrite(9,LOW);
  pinMode(17, OUTPUT);
  digitalWrite(17,HIGH);



  while((int)inval != 32){                          // INPUTS to IC
    Serial.println(" ");


    if((int)inval == 48){       // 0
      pinMode(inpin, OUTPUT);
      digitalWrite(inpin,LOW);
      delay(100);}

    if((int)inval==49){       // 1
      pinMode(inpin, OUTPUT);
      digitalWrite(inpin,HIGH);
      delay(100);}

    Serial.print("");
    Serial.print("Pin ");
    Serial.print(inpin);
    Serial.print(" sets: " );
    Serial.println(inval);

    up = digitalRead(26);
    if(up==1){Menu();}

    inpin = inpin + 1;
    inadress = inadress + 1;
    inval = Read_Eeprom(inadress);}

while((int)clockval != 32){      //Clock once all inputs have been done
    Serial.println("");
    if((int)clockval == 67){
      Serial.println("Triggering Clock");
      pinMode(clockpin, OUTPUT);
      digitalWrite(clockpin,LOW);                    // set it  LOW
      delay(100);
      digitalWrite(clockpin,HIGH);                   // set it  HIGH
      delay(100);}

    up = digitalRead(26);
    if(up==1){Menu();}

    clockpin = clockpin + 1;
    clockadress = clockadress + 1;
    clockval = Read_Eeprom(clockadress);}

  while((int)outval !=32){                          // OUTPUTS from IC
    Serial.println(" ");
    if(((int)outval == 76) || ((int)outval == 72)){
      pinMode(outpin, INPUT_PULLUP);
      int response = digitalRead(outpin);
      Serial.print("Response at outpin ");
      Serial.print(outpin);
      Serial.print(" has been: ");
```

```
        Serial.println(response);
        Serial.print("And it should be: ");
        Serial.println(outval);
        //convert response to L H
        int convertoutval = 1;
        if((int)outval == 76){
          convertoutval = 0;
        }
        if((int)outval == 72){
          convertoutval = 1;
        }
        Serial.print("response: ");
        Serial.println(response);
        Serial.print("convertoutval: ");
        Serial.println(convertoutval);

        if(response != convertoutval){  //Response = IC ; Outval = Eeprom
          correctIc = false;
          Serial.println("FALSE at: ");
          Serial.println(outadress);
          break;}
        }
      up = digitalRead(26);
      if(up==1){Menu();}

      outpin = outpin + 1;
      outadress = outadress +1;
      outval = Read_Eeprom(outadress);}
    if (correctIc == false){
      break;}
    adress = inadress + 1;                    //29+1=30
    check_val = Read_Eeprom(adress);
    Serial.println(" ");
    Serial.println(" ");
    Serial.println(adress);
    Serial.println(check_val = Read_Eeprom(adress));
    Serial.println(" ");
    Serial.println(" ");
    Serial.println(" ");}
  Serial.println("End of check pins16 and reset");
  int pinreset = 2;
  while(pinreset<=17){
    digitalWrite(pinreset,LOW);
    pinreset =  pinreset +1;
  }
  return correctIc;}
```

o A.3 Funció Anàlisi d'IC

```
void Analyze_IC(int numpins){
  int up = 0;
  digitalWrite(35,LOW);
  digitalWrite(36,HIGH);
  digitalWrite(37,LOW);
  if(numpins == 14){
    Display_Lcd("Pin state:",0);
    delay(1000);
    while(true){
    char Pins[16]={'\0'};
    char Analogpins[] = {A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13};
    for(int i = 0; i<numpins;i++){
      int pinval = analogRead(Analogpins[i]);
      Serial.print("analogval: ");
      Serial.println(pinval);
      if(pinval > 1000){
        Pins[i]='1';}
      if(pinval < 50){
        Pins[i]='0';}
      if((pinval < 1000)&&(pinval > 50)){
        Pins[i]='x';}

      up = digitalRead(26);
      if(up == 1){Menu();}
    }
    Serial.print("Pins ");
    Serial.println(Pins);
    delay(500);
    Display_Lcd_Arrow("12345678901234",Pins,0,0);}}
  if(numpins == 16){
    Display_Lcd("Pin state:",0);
    delay(1000);
    while(up == 0){
    char Pins[18]={'\0'};
    char Analogpins[] =
{A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15};
    for(int i = 0; i<numpins;i++){
      int pinval = analogRead(Analogpins[i]);
      Serial.print("analogval: ");
      Serial.println(pinval);
      if(pinval > 1000){
        Pins[i]='1';}
      if(pinval < 50){
        Pins[i]='0';}
      if((pinval < 1000)&&(pinval > 50)){
        Pins[i]='x';}
      up = digitalRead(26);
      if(up == 1){Menu();}
    }
    Serial.print("Pins ");
    Serial.println(Pins);
    delay(500);
    Display_LcdTwowords("1234567890123456",Pins);}
    }}
```

o A.4 Funció Veure data

```
void View_Data(){
 digitalWrite(35,HIGH);
 digitalWrite(36,HIGH);
 digitalWrite(38,LOW);
 digitalWrite(39,HIGH);
 int adress = 0;
 char val = Read_Eeprom(adress);
 char numpins[3]={'\0'};

 int select = 0;
 int up = 0;
 int down = 0;
 int left = 0;
 int right = 1;

while(select == 0){
 while(select == 0){
  if(right == 1){
    while(val != '$'){                  // while IC not found
    adress = adress + 1;
    val=Read_Eeprom(adress);}
    char Name[6]={'\0'};
    int index = 0;
    while(val != ' '){
       adress = adress + 1;
       Name[index] = (char)(Read_Eeprom(adress));
       index = index + 1;
       val=Read_Eeprom(adress);}

     Serial.println("Next IC:");
     numpins[0]=(char)(Read_Eeprom(adress+1));
     numpins[1]=(char)(Read_Eeprom(adress+2));
     Display_Lcd_Arrow("Select IC",Name,2,0);
     delay(1000);}

  up = digitalRead(26);
  if(up == 1){Menu();}
  right = digitalRead(29);
  select = digitalRead(25);}

  //Once selected IC
  digitalWrite(35,LOW);
  digitalWrite(36,LOW);
  adress = adress + 5;
  char data_val = Read_Eeprom(adress);
  while(((int)data_val != 36)&&((int)data_val != 38)){
    int data_index = 0;
    char Data[18]={'\0'};
    while(data_val != ' '){
       Data[data_index] = (char)(Read_Eeprom(adress));
       adress = adress + 1;
       data_index = data_index + 1;
       data_val=Read_Eeprom(adress);}
```

```
    Display_Lcd_Arrow(Data,"",2,1);
    delay(1000);
    right = 0;
    while(right == 0){right = digitalRead(29);}
    adress = adress + 1;
    data_val = Read_Eeprom(adress);}

    Display_Lcd("Press left to    3. View Data",0);
    digitalWrite(38,HIGH);
    left = 0;
    while(left == 0){left = digitalRead(28);}}
    View_Data();}
```

o A.5 Funció Reemplaçar IC

```
void Replace_IC(){
  Display_Lcd_Arrow("Select value of pins: ","",0,0);
  delay(2000);
  int select = 0;
  int up = 0;
  int down = 0;
  int left = 0;
  int right = 0;

  char Pins[18]={'\0'};
  Serial.println(Pins);
  int pos = 0;
  int Clock = 0;
while(pos<16){
  digitalWrite(36,HIGH);
  digitalWrite(37,HIGH);
  int i = 0;
  select = 0;
  while(select == 0){
    down = 0;
    if(i == 0){
      while(down == 0){
        Display_Lcd_Arrow_Pos("1234567890123456",Pins,1,1,pos);
        delay(200);
        Display_Lcd_Arrow_Pos("1234567890123456",Pins,0,1,pos);
        delay(200);
        Display_Lcd_Arrow_Pos("1234567890123456",Pins,1,1,pos);
        down = digitalRead(27);
        up = digitalRead(26);
       if(up==1){Menu();}}

    }
    while(down == 0){
      down = digitalRead(27);
      select = digitalRead(25);
      if (select == 1){
        i = i-1;
        break;}
      }
```

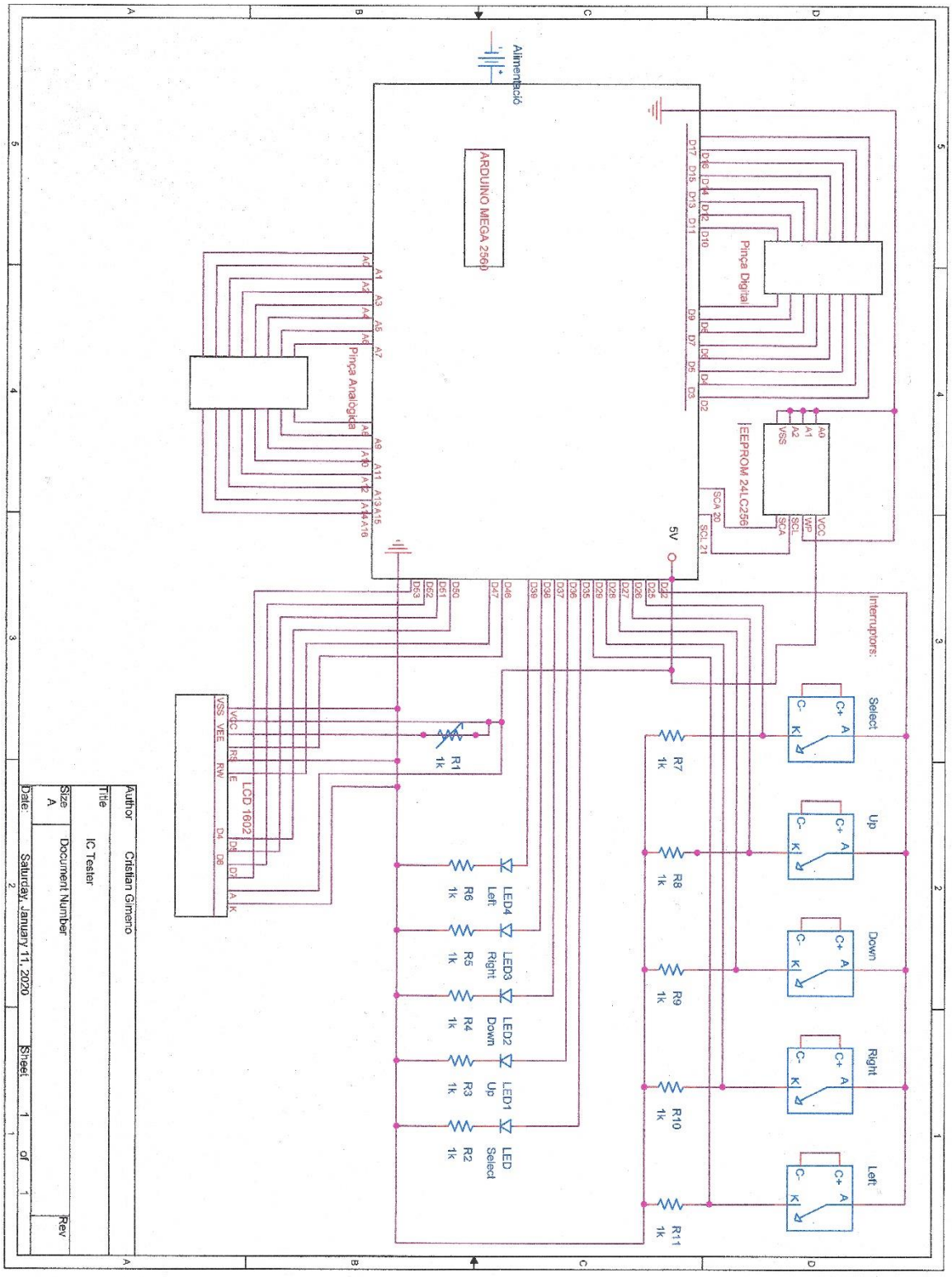
```
    i = i+1;
    if(i == 1){Pins[pos] = '1';}
    if(i == 2){Pins[pos] = '0';}
    if(i == 3){Pins[pos] = 'H';}
    if(i == 4){Pins[pos] = 'L';}
    if(i == 5){Pins[pos] = 'V';}
    if(i == 6){Pins[pos] = 'G';}
    if(i == 7){Pins[pos] = 'C'; Clock = pos;}
    if(i == 8){Pins[pos] = 'X';}
    if(i == 9){i=0;}
    delay(500);

    digitalWrite(35,HIGH);
    down = digitalRead(27);
    Display_Lcd_Arrow("1234567890123456",Pins,1,1);
    if(pos == 15){Display_Lcd_Arrow("1234567890123456",Pins,0,1);}
    }
    digitalWrite(35,LOW);

  if(i != 7){Pin_Replacement(pos,i);}
  if(i == 7){int Clock = pos;}
  up = digitalRead(26);
  if(up == 1){Menu();}
  pos = pos + 1;}
  delay(500);
  if(Clock != 0){
    Pin_Replacement(Clock,7);}
  Display_Lcd("Set Replacement of: ",1);
  delay(1000);
  Display_Lcd(Pins,1);
}
void Pin_Replacement(int pin, int val){

  if((val == 1)||(val == 3)||(val == 5)){       //HIGH
    pinMode(pin+2, OUTPUT);
    digitalWrite(pin+2,HIGH);
    delay(100);}

  if((val == 2)||(val == 4)||(val == 6)){       //LOW
    pinMode(pin+2, OUTPUT);
    digitalWrite(pin+2,LOW);
    delay(100);}

  if(val == 7){                                 //CLOCK
    pinMode(pin+2, OUTPUT);
    digitalWrite(pin+2,LOW);
    delay(10000);
    digitalWrite(pin+2,HIGH);
    delay(100);}
  }
```

o A.6 Funció Write_Data

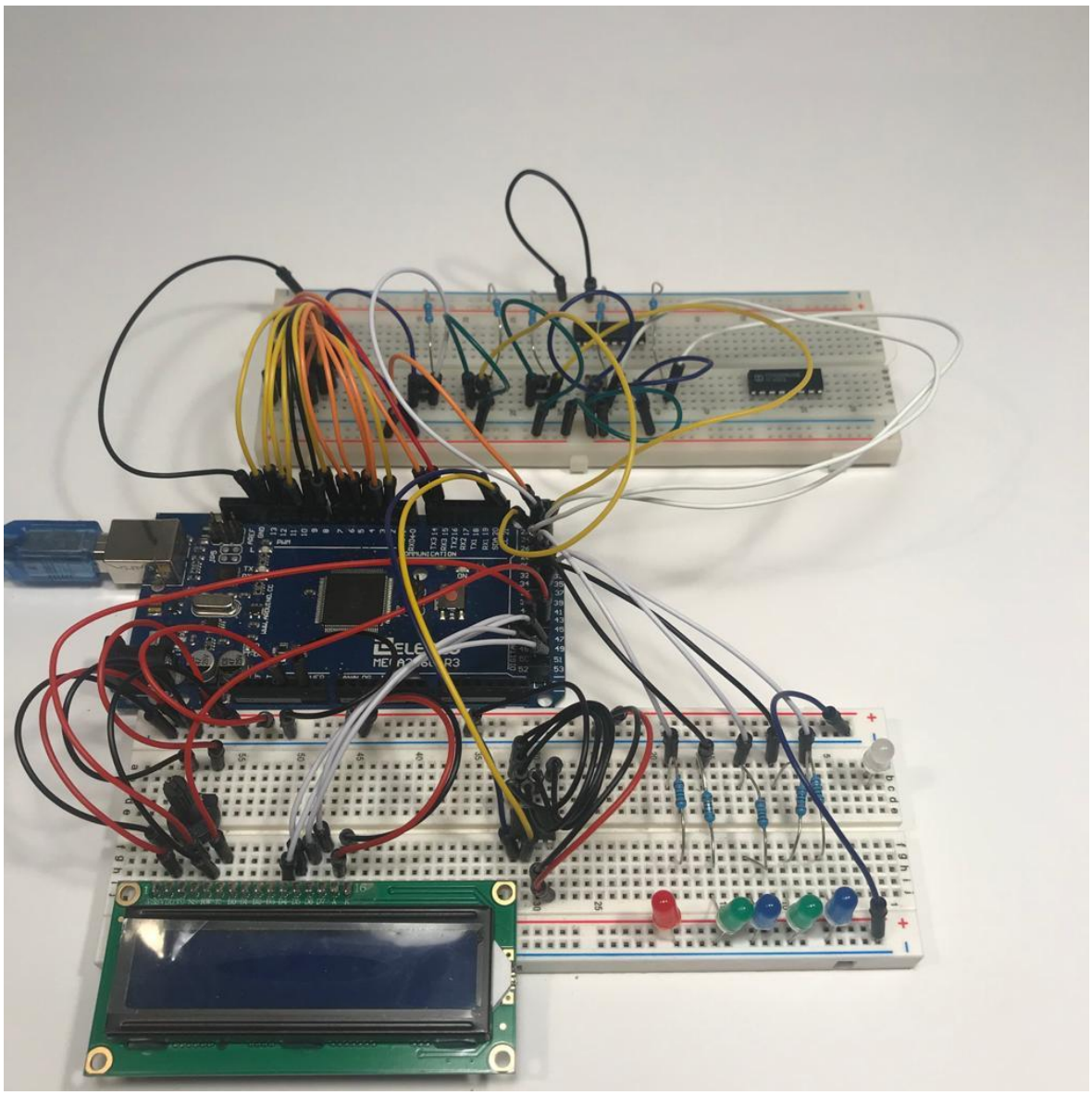
```
#include "Wire.h"
#define Eeprom_Adress 0x50

void setup() {
  Wire.begin();
  Serial.begin(9600);
  char val = 'a';
  int adress = 0;
  Writing_Data(adress,val);}

void Write_Eeprom(int adress, char val) {
  Wire.beginTransmission(Eeprom_Adress);    //Start tranmission
  Wire.write((int)(adress >> 8));
  Wire.write((int)(adress & 0xFF));
  Wire.write(val);
  Wire.endTransmission();
  delay(5);
}

void Writing_Data(int adress,char val){
  int x = 0;
  char data_trial[] = {"0004 $4009 16  VH0H0H0G0H0HX0HV VL1L1L1G1L1LX1LV
$4015 16  CLLLL10GCLLLL10V CLLLH01GCLLLH01V CLLHH01GCLLHH01V
CLHHH01GCLHHH01V CHHHL00GCHHHL00V $4009 16  VH0H0H0G0H0HX0HV
VL1L1L1G1L1LX1LV $74595 16  XXXXXXXGX00000XV LLLLLLLGL10000LV
LLLLLLLGL1C001LV LLLLLLLGL1C001LV LLLLLLLGL1C001LV LLLLLLLGL1C001LV
HHHLLLLGL10C01HV $4013 14  LHC100G001CHLV HLC001G100CLHV LHC000G000CHLV
HLC010G010CLHV $4002 14  L00010G01100LV L00010G01000LV L00010G01111LV
L00110G00011LV L11000G01111LV L01000G00100LV L11110G01111LV
H00000G00000HV L10100G01010LV L01010G00101LV $4010 16  VH1H1H1G1H1HX1HV
VL0L0L0G0L0LX0LV $40106 14  0H0H0HGH0H0H0V 1L1L1LGL1L1L1V $4011
14  00HL11G11LH00V 10HH10G10HH10V 01HH01G01HH01V 11LH00G00HL11V $4012
14  H0000XGX0000HV H1010XGX1010HV H0101XGX0101HV L1111XGX1111LV &"};
  Serial.println("Sizeof(data_trial)");
  Serial.println(sizeof(data_trial));
  while(x<sizeof(data_trial)){
    Serial.println(x);
    val = data_trial[x];     //reads
    Serial.print("I recieved and write: ");    //prints
    Serial.println(val);
    Write_Eeprom(adress, val);
    adress = adress + 1;
    x=x+1;}
  Serial.println("Adress_data_trial ends in: ");
  Serial.println(adress);
  Serial.println(val);
}
```

• ANNEX B: PROTOTIPATGE

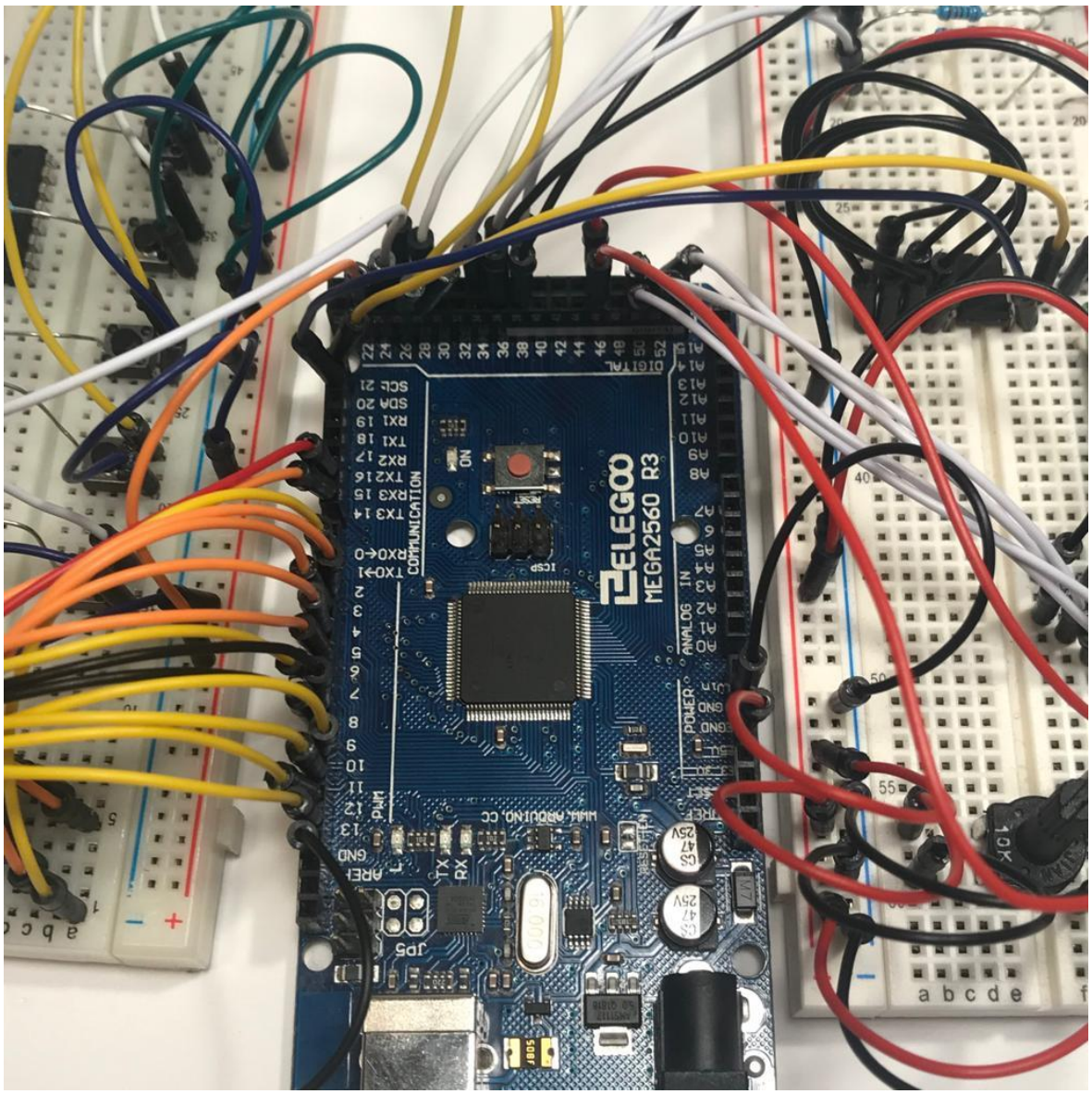
○ B.1 Esquemàtic de connexions



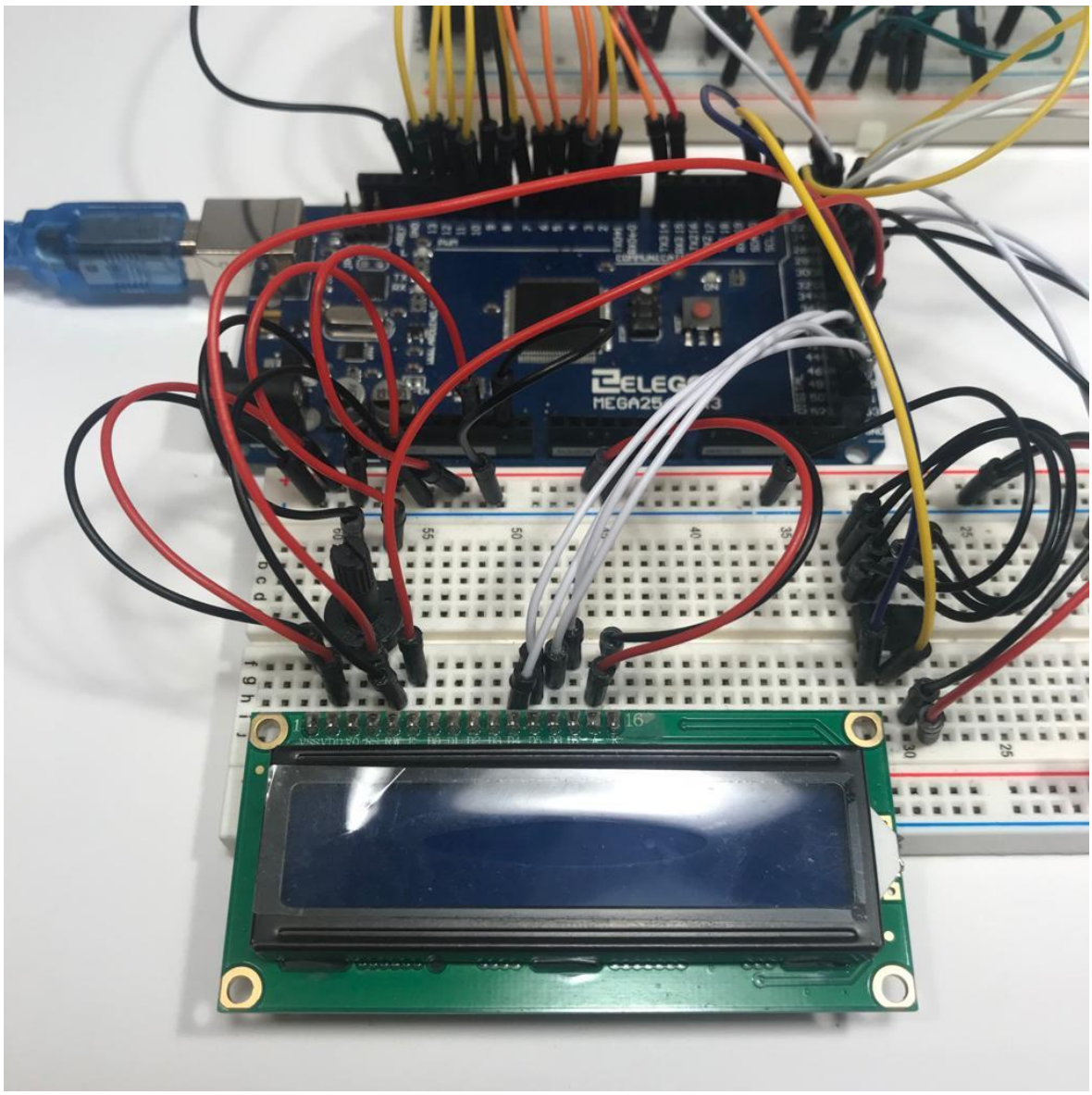
○ B.2 Conjunt



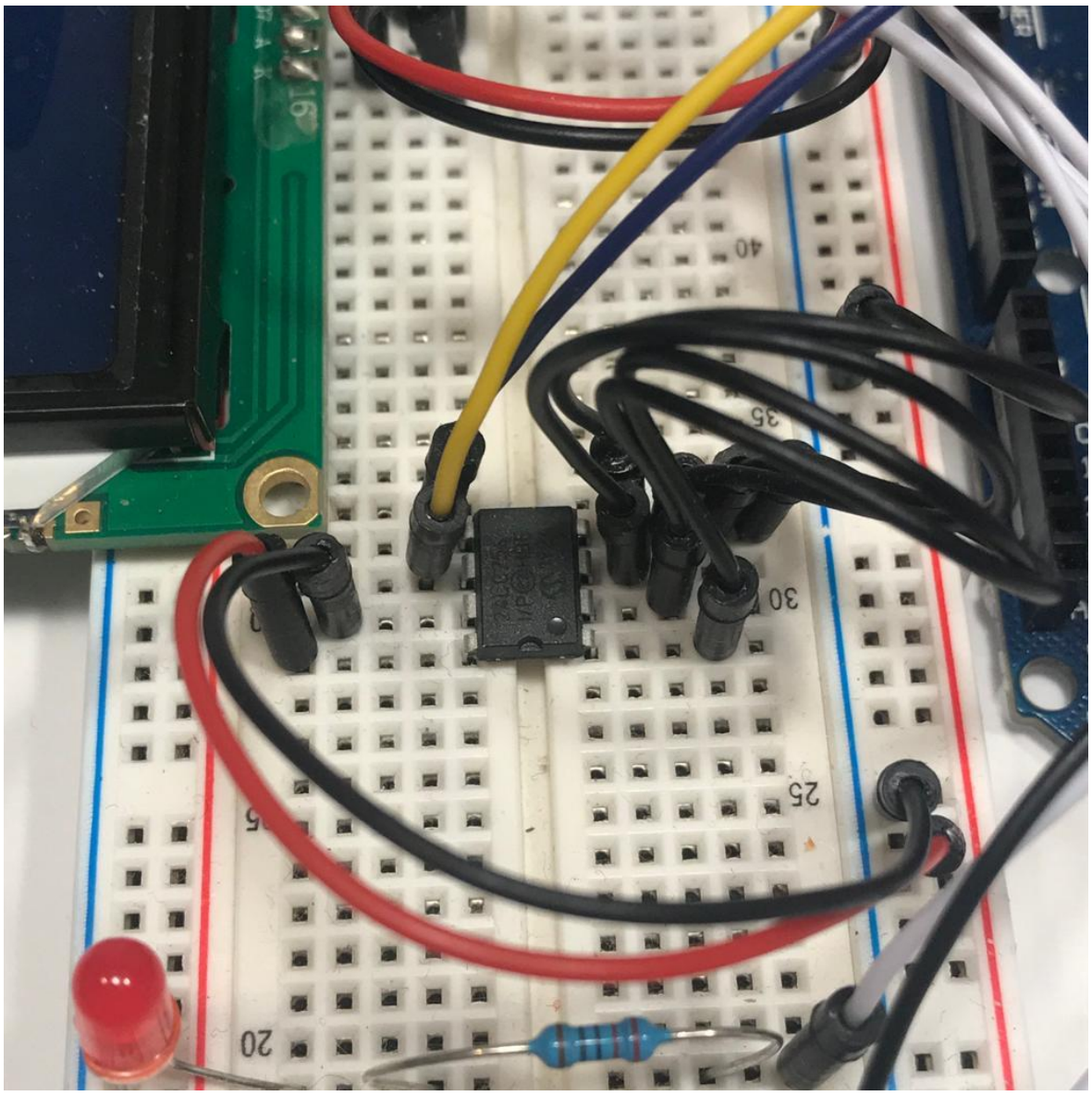
○ B.3 Arduino


ELEGOO
MEGA2560 R3
DIGITAL
ANALOG IN
POWER
COMMUNICATION
PWM
ON
ICSP
TX
RX
a b c d e

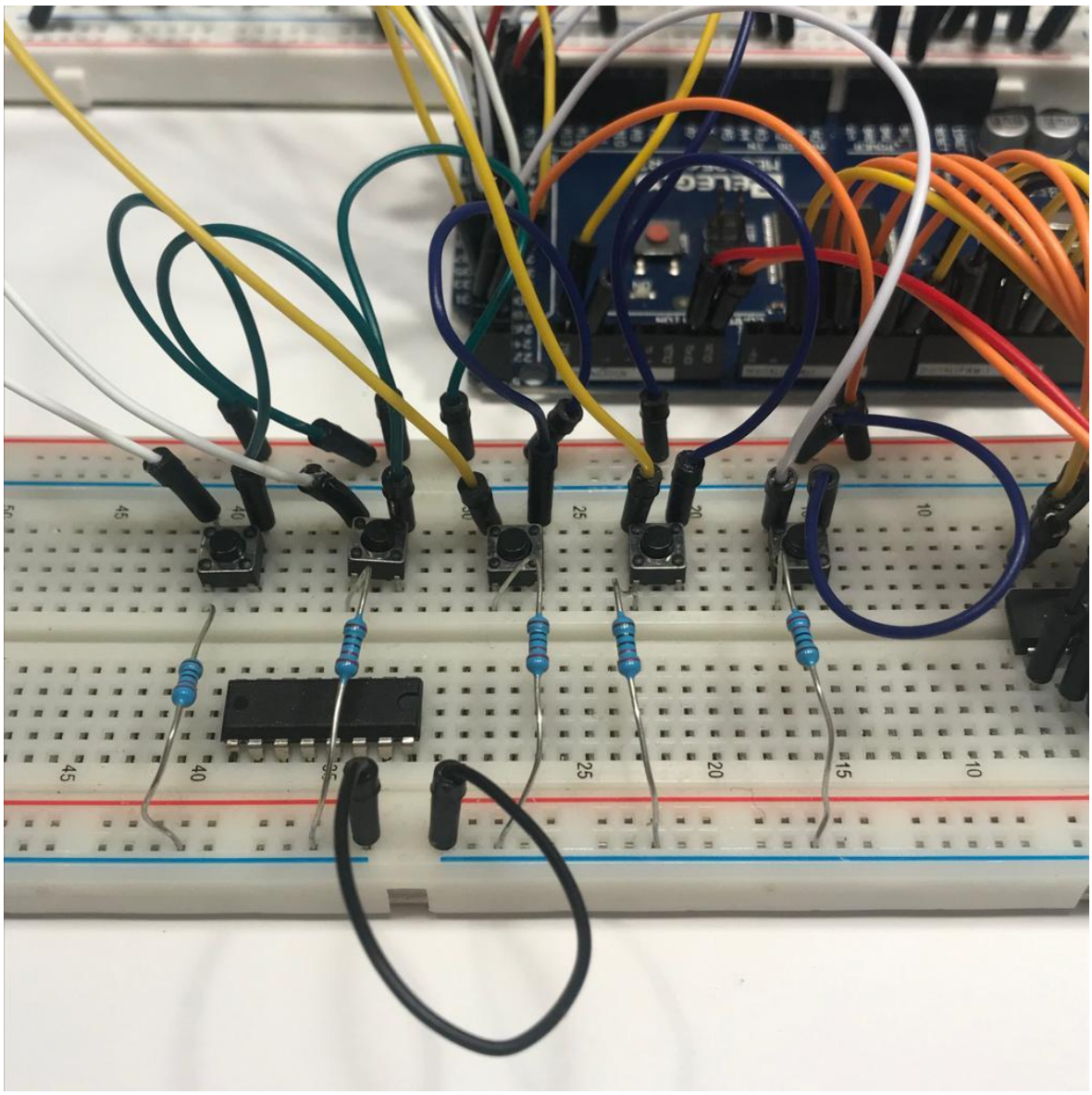
- **B.4 LCD 1602**



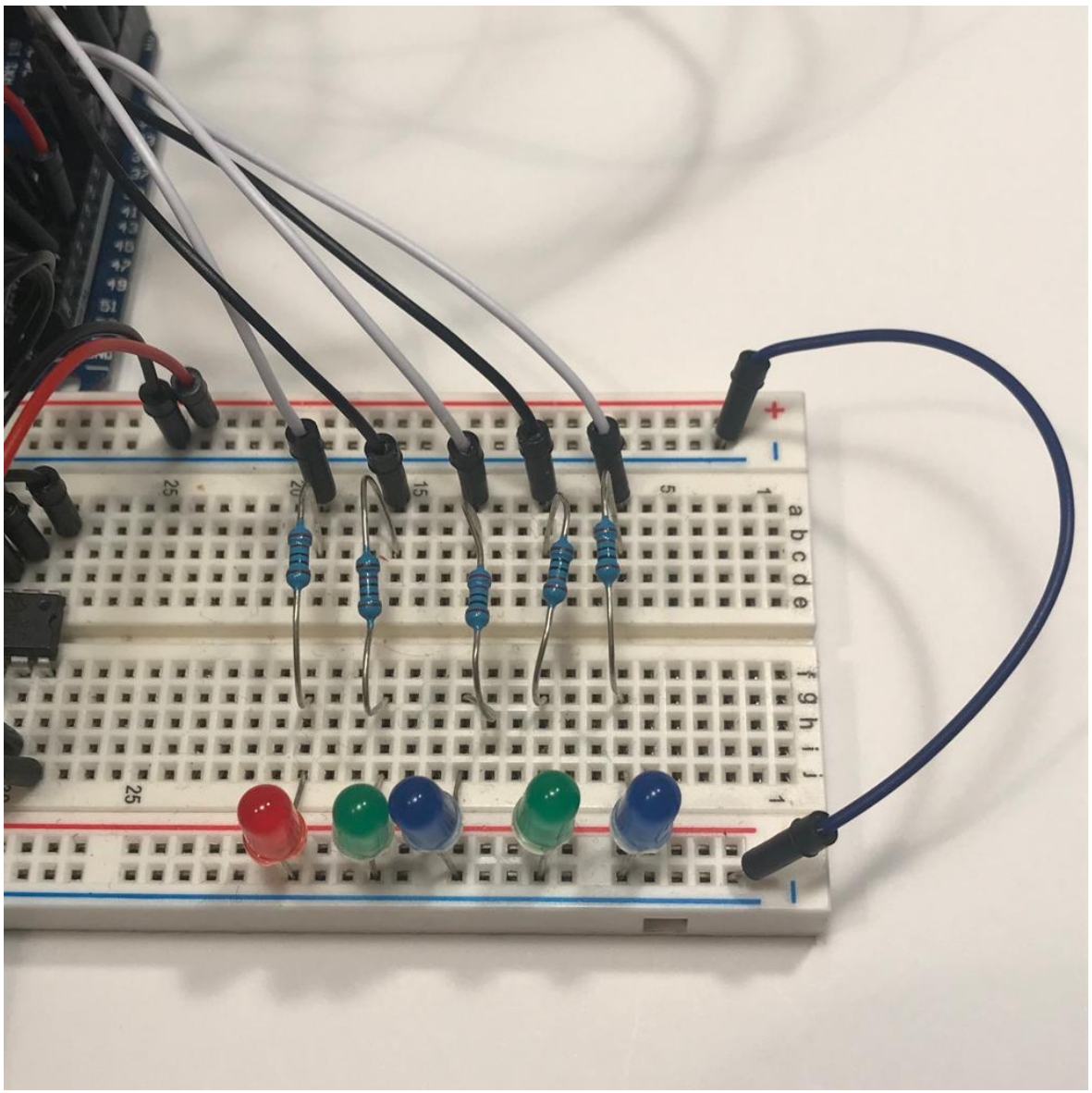
○ B.5 EEPROM 24LC256



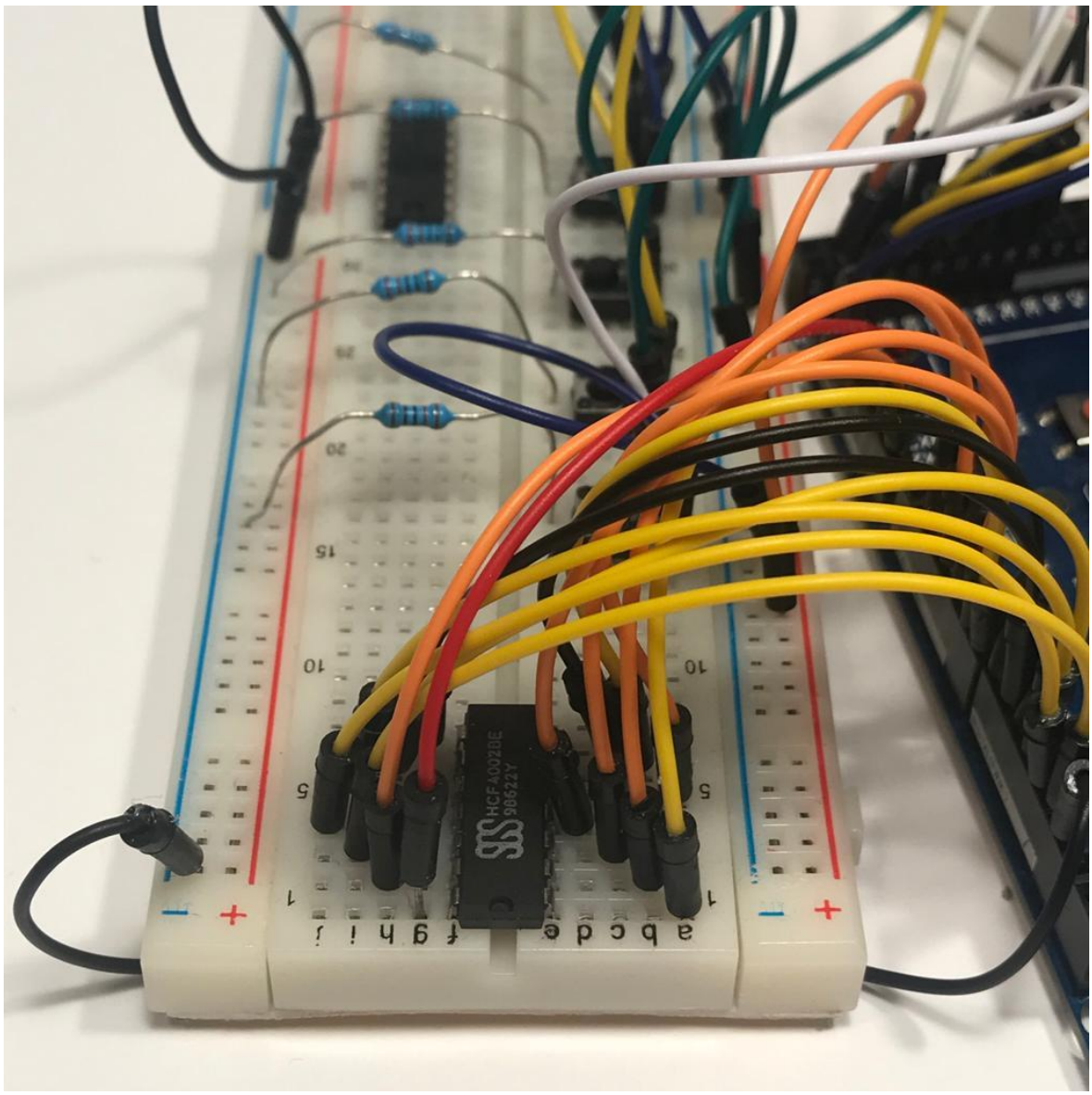
- **B.6 Control**



- **B.7 LED**

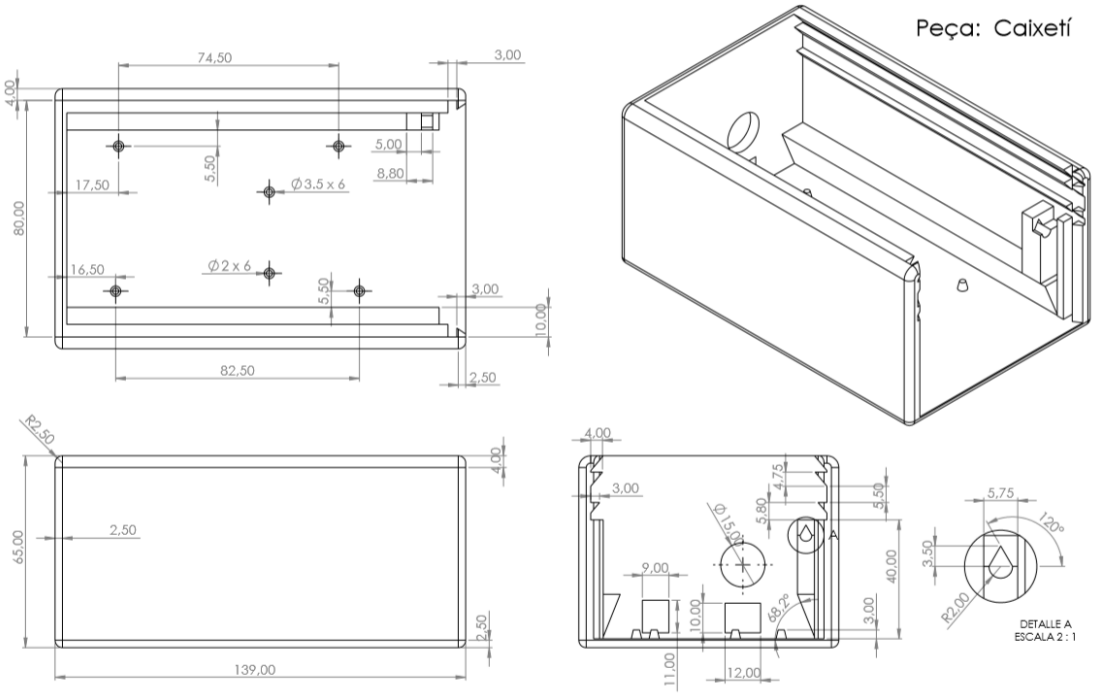


- **B.8 Pinça**

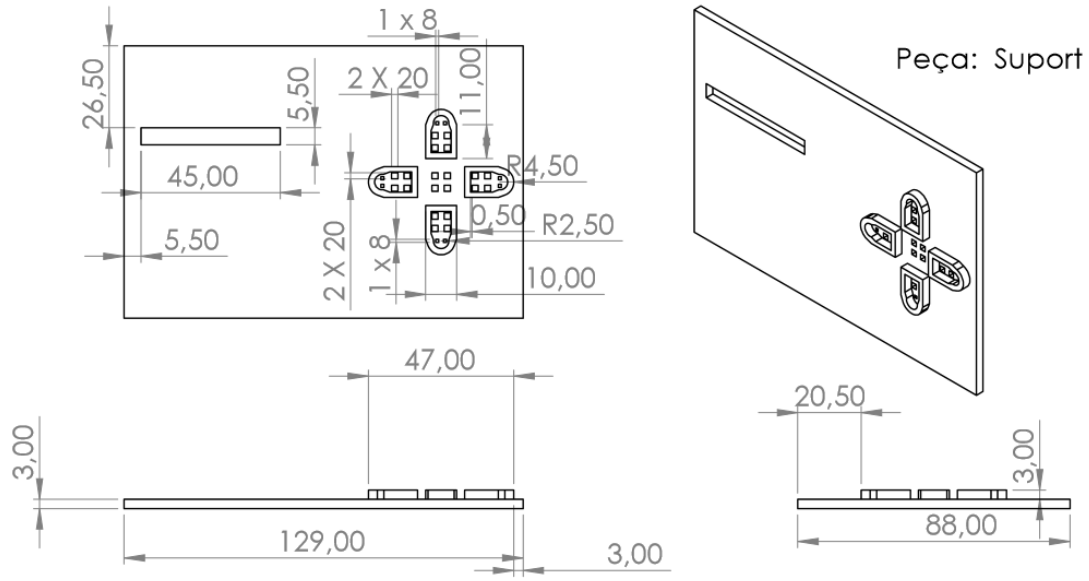


- # ANNEX C: PLÀNOLS CARCASSA

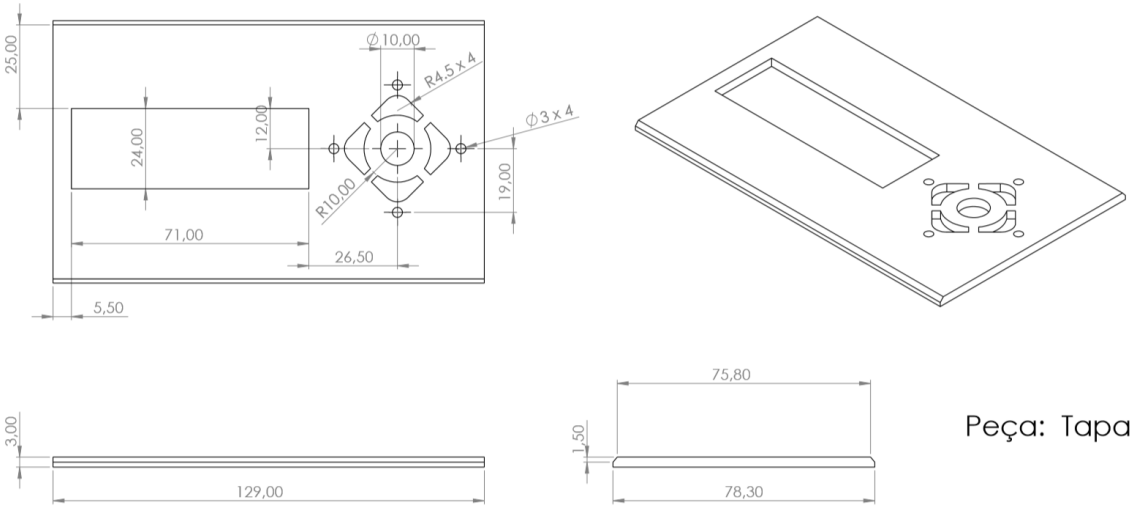
○ C.1 Plànol de peça Caixetí



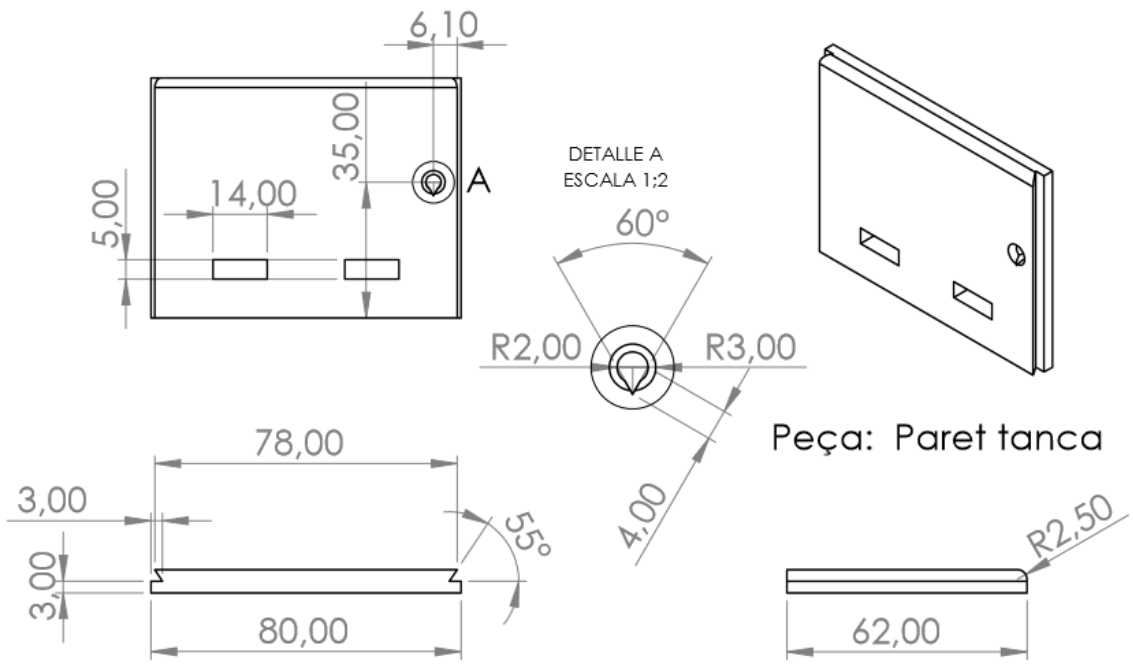
○ C.2 Plànol de peça Suport



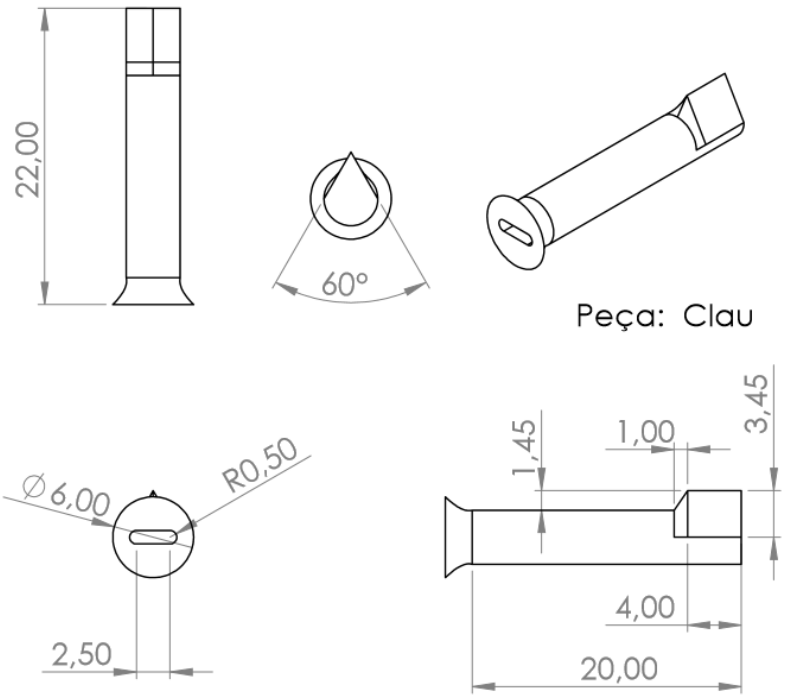
○ C.3 Plànol de peça Tapa



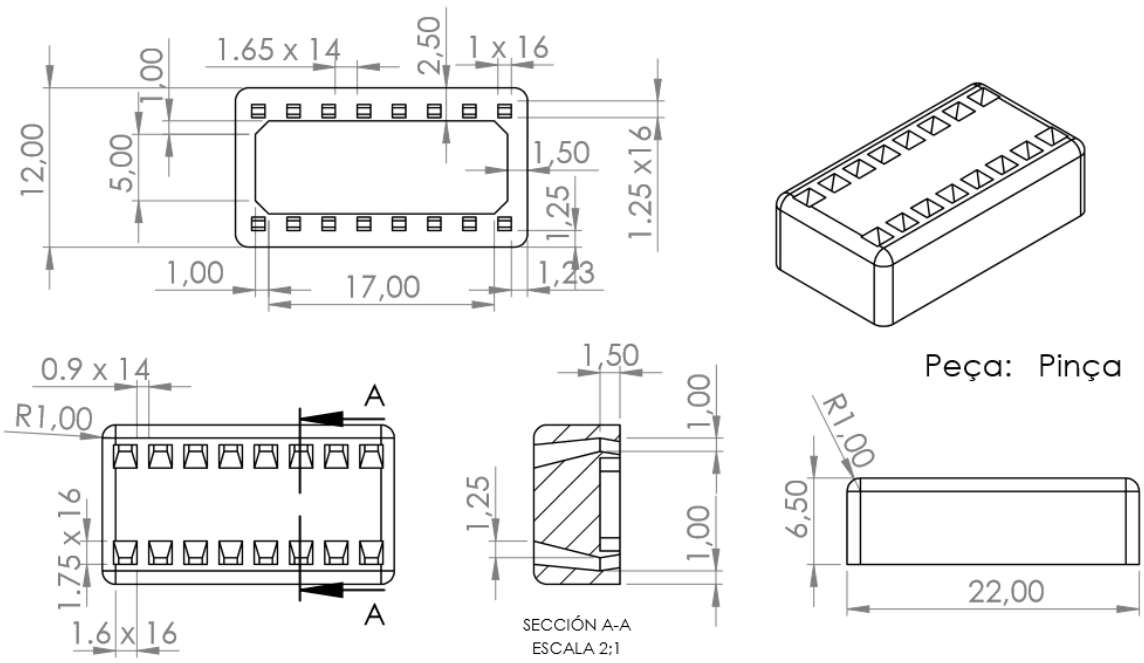
○ C.4 Plànol de peça Paret Tanca



○ C.5 Plànol de peça Clau



○ C.6 Plànol de peça Pinça



- **ANNEX D: GUIA DE CONSTRUCCIÓ**

Introduction

Welcome to the IC Tester Assembly guide.

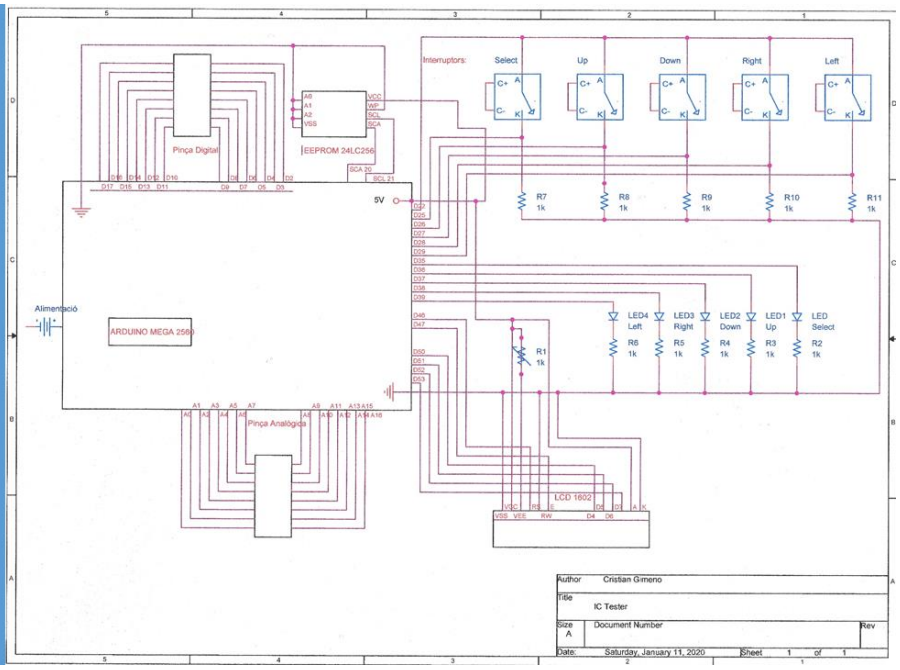
Through this 16 simple steps you will be able to assembly your own IC Tester and be ready to test all the integrated circuits within the 7400 and 4000 series.

Before starting the assembly, it is necessary to have all the electronic compounds and the IC Tester case pieces, these ones you can get them by 3D Printing with the STL documents attached to the guide.

Let's start!!!

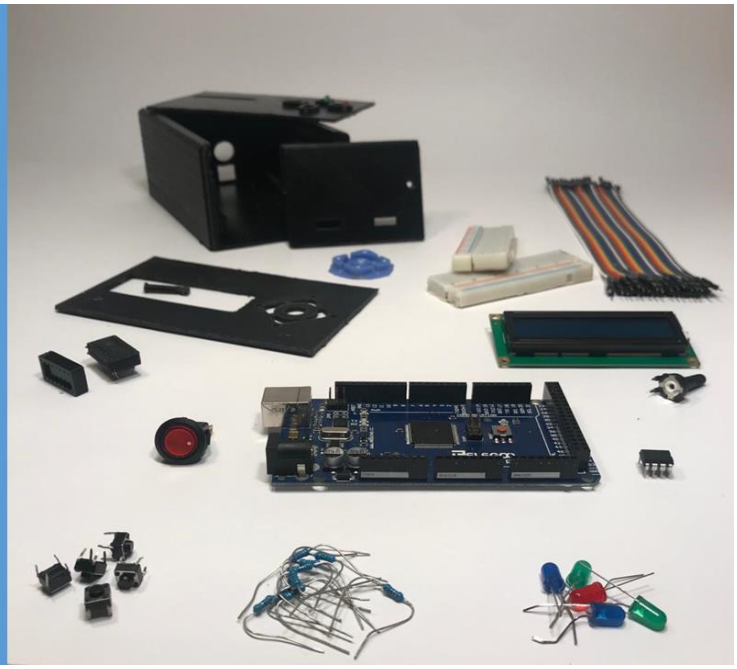
Difficulty: Moderated Time: 90 Minutes Steps: 16

Schematic



Step 1 – Materials (13)

- IC Tester Case
- IC Tester Clamp x2
- Arduino (Elegoo) Mega 2560
- EEPROM 24LC256
- LCD 1602
- Potentiometer 10K
- Control buttons
- Button Switch
- Power Switch
- 10 Resistors 1K
- LED 3mm
- Pieces of Protoboard
- Cabling H-M M-M



Step 2 – Placing Arduino

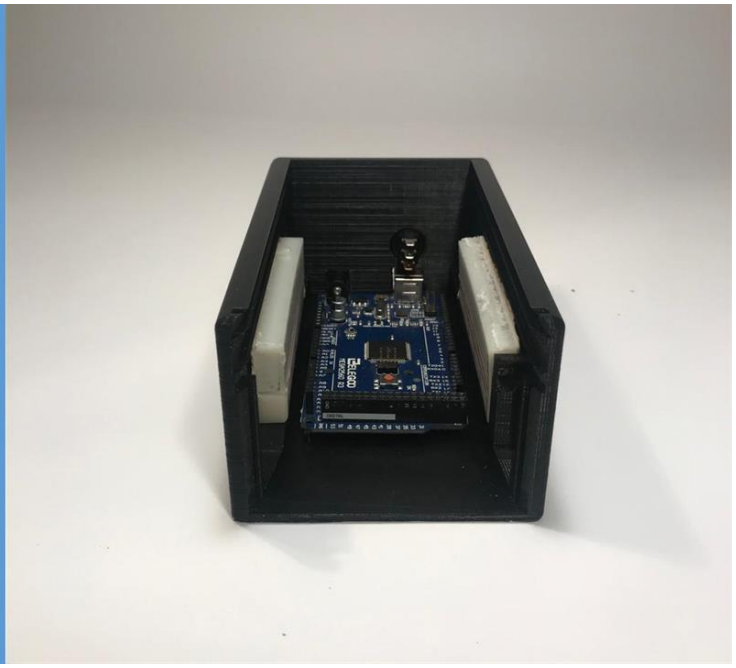
- Place Arduino inside the base of the case.
- Be sure to adapt correctly Arduino's power and data entrances to the back wall of the case.

Step 3 – Placing Power Switch

- Place Power Switch in the back of the case.

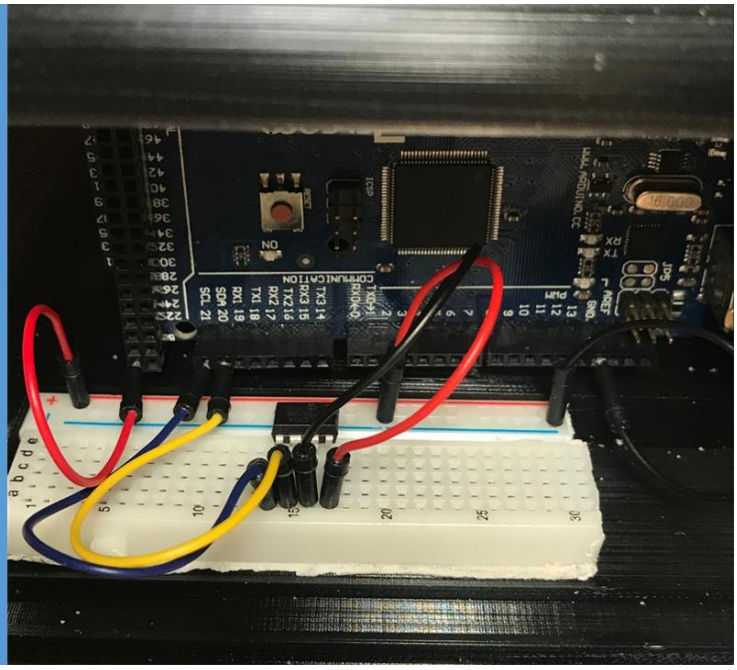
Step 4 – Protoboard

- Attach Protoboard to both lateral walls of the case.
- Place the protoboard polarized section on the bottom, under the non polarized one.
- Power up the lines of the protoboard connecting with Arduino through the power switch.



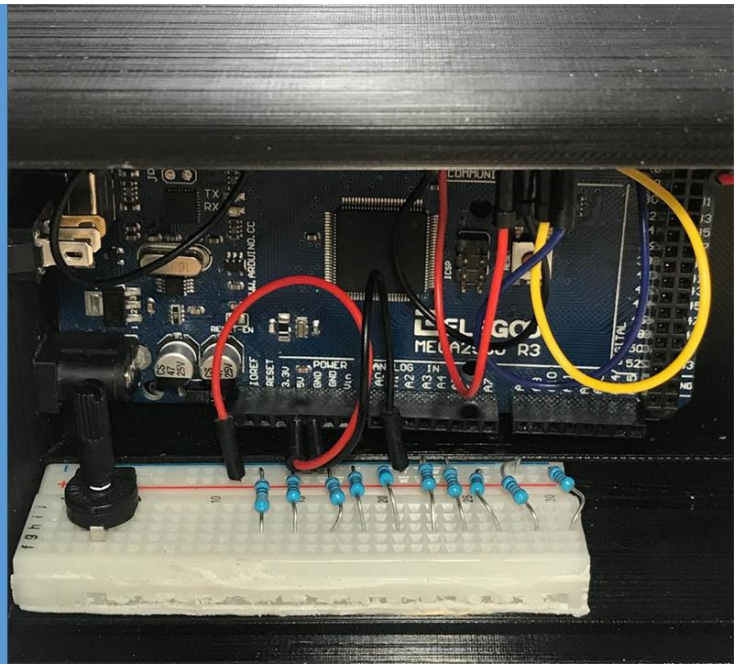
Step 5 – EEPROM

- Attach EEPROM to the protoboard.
- Connect it to the Arduino's respective ports (SCL, SCA) as shown on the Schematic.



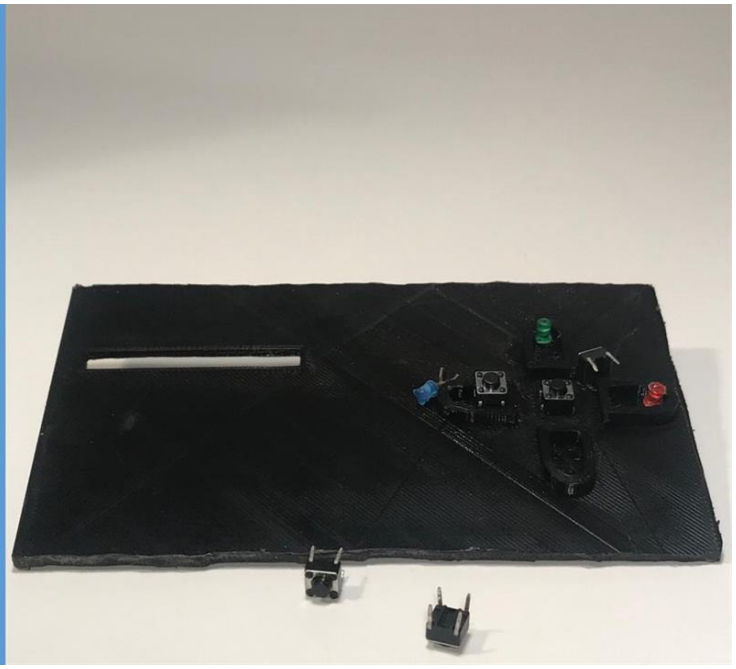
Step 6 – Resistors

- Add all the resistors and the potentiometer to the other protoboard wall.
- Leave some space between resistors and power up the protoboard lines with 5V and Ground.



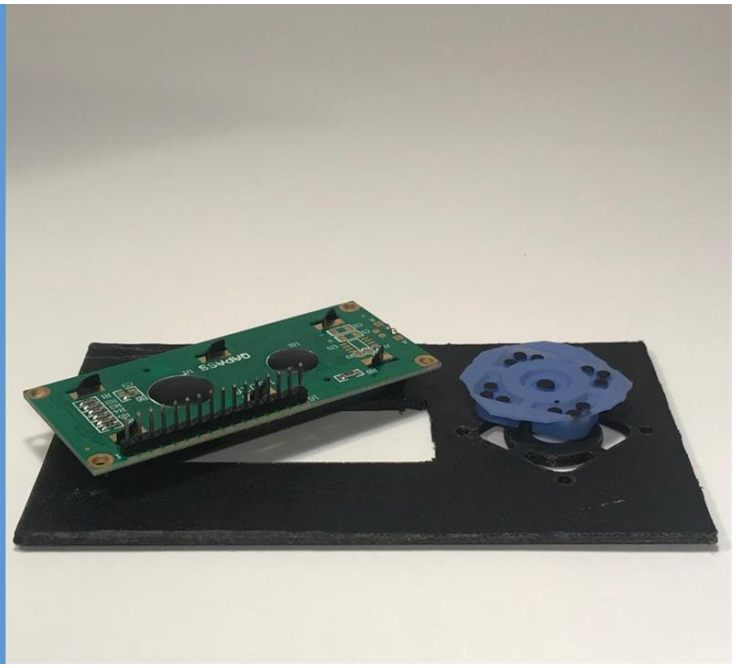
Step 7 – Switches and LEDs

- Place the 5 switches and 5 LEDs on the middle rack.
- Use the designed gaps to stabilized all the compounds.



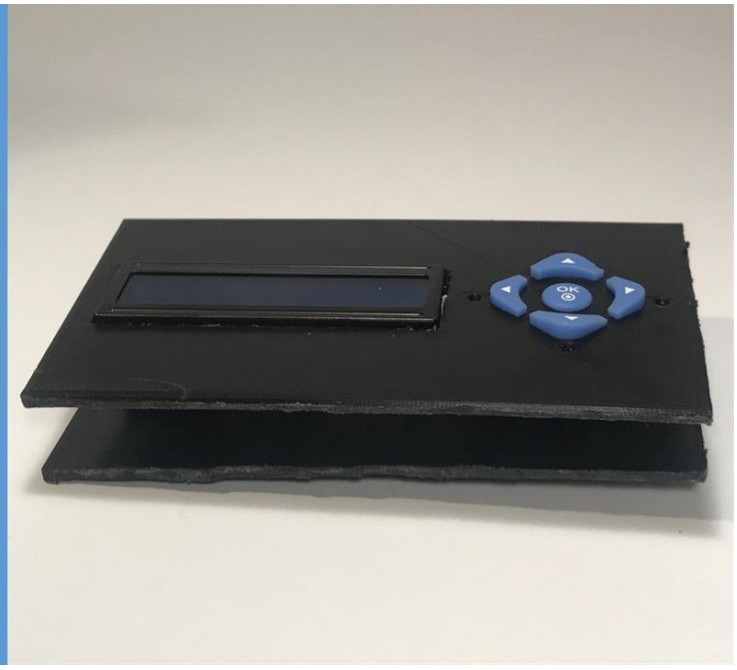
Step 8 – LCD and Control

- Introduce LCD and control buttons into the upper rack.
- Place LCD according to the middle rack geometry.



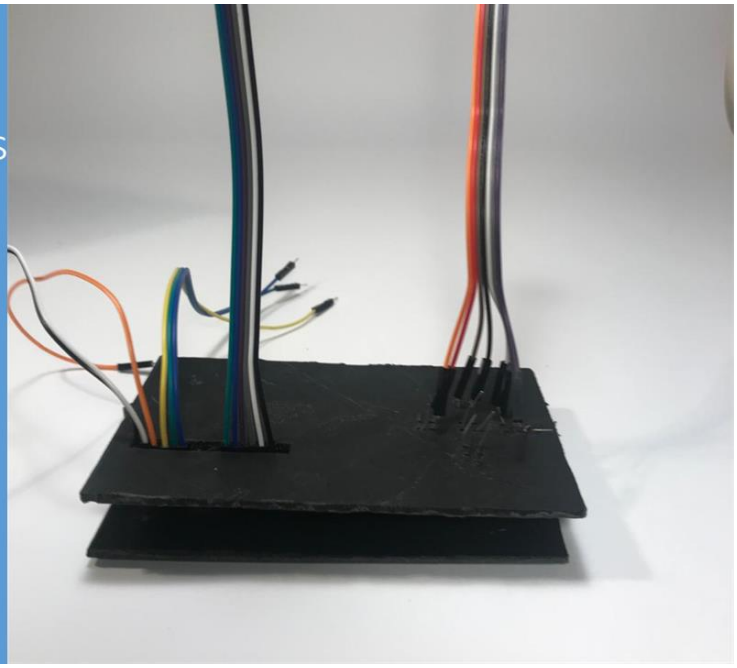
Step 9 – Racks

- Face both racks to adjust the LEDs and control buttons before making connections.



Step 10 – Connections

- Connect LCD to Arduino pins as shown on the Schematic, remember to connect the third LCD pin to the potentiometer before getting into the microcontroller.
- Connect switches and LEDs to each resistors through the protoboard, then connect them into the respective Arduino pin as shown on the Schematic.



Step 11 – Clamp Cabling

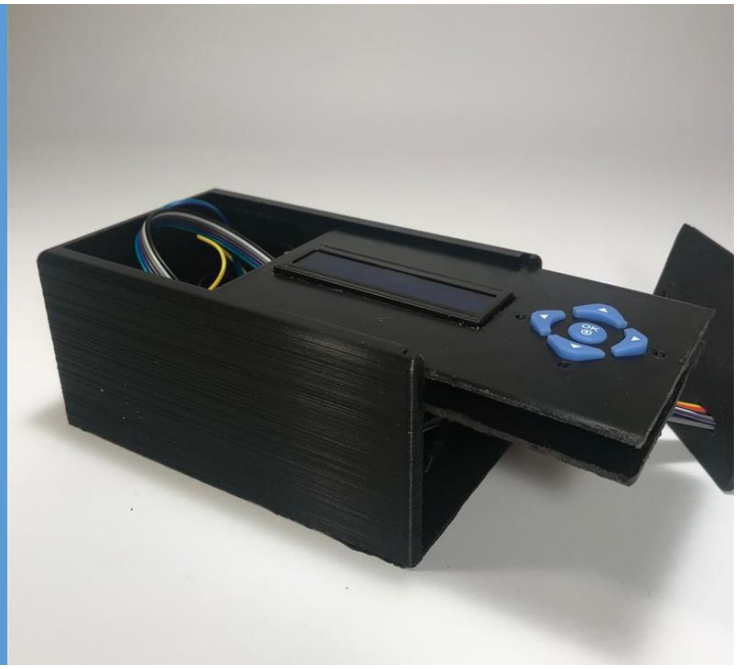
- Putt all the M-M cabling through the entrances, 16 connectors for each entrance, 32 in total.
- Left entrance will be the analog and righ entrance the digital one.

Step 12 – Cabling

- Connect 32 M-M wires into each Arduino pin as shown on the Schematic.
- Remember left cabling is analog and right is digital.

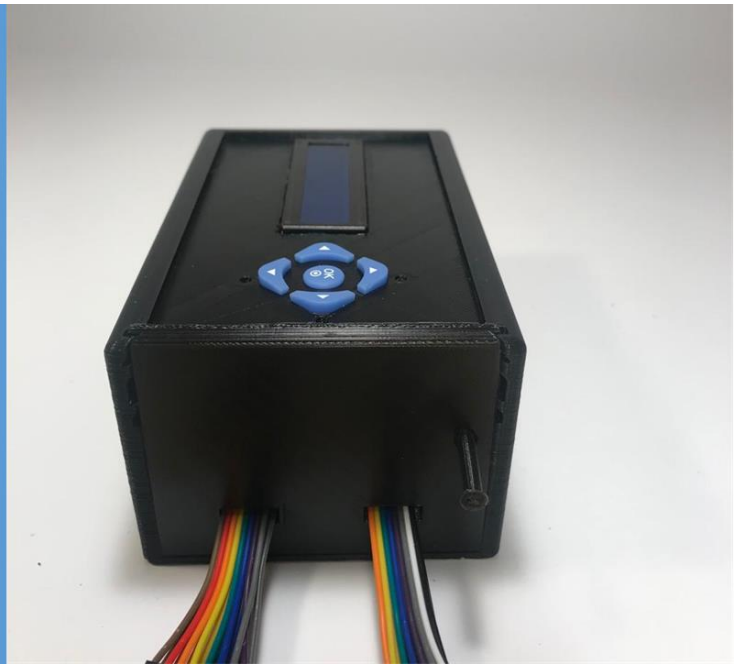
Step 13 – Placing Racks

- Once everything connected, place both racks at the same time on the case supports until reaching the back case wall.
- Next, introduce from top to bottom the last wall with the two entrances.



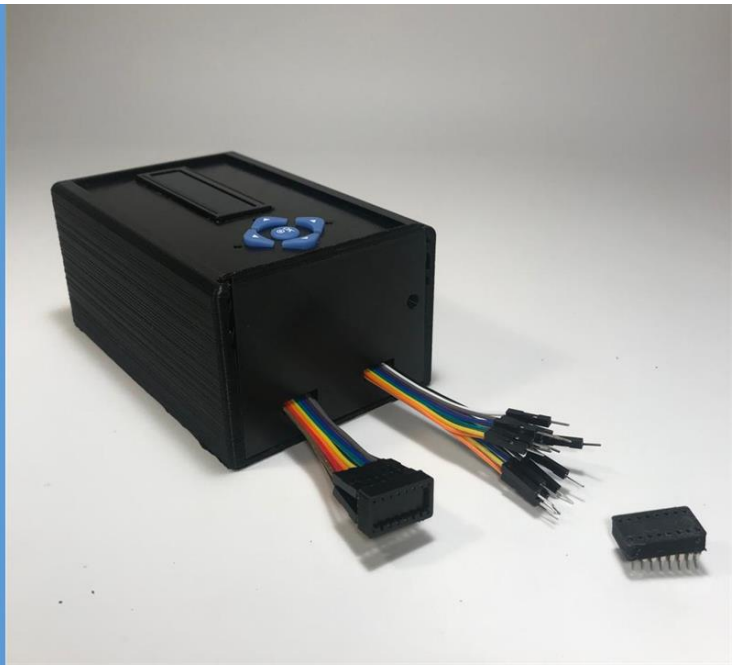
Step 14 – Insert the key

- Insert the small triangular key through the entrance wall with the triangle facing down.
- Once half in, turn the key 180º, then press until reaches the end.



Step 15 – Clamps

- Connect each clamp with analog and digital cabling.



Step 16
Ready to test!

Author
Cristian Gimeno